VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Analýza síťových anomálií pomocí síťových statistik NetFlow

Network Anomaly Analysis with NetFlow Network Statistics

2015/2016 **Václav Stefek**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Václav Stefek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T059 Mobilní technologie

Téma: Analýza síťových anomálií pomocí síťových statistik NetFlow
Network Anomaly Analysis with NetFlow Network Statistics

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem diplomové práce je analýza síťových anomálií s využitím síťových statistik protokolu NetFlow. Výsledky analýz budou testovány pomocí systémů Bro IDS a Suricata IDS.
1. Zachycení a rozbor dat v souboru pcap.
2. Popis protokolu NetFlow.
3. Analýza síťových anomálií.
4. Testování systému Bro IDS.
5. Testování systému Suricata IDS.
6. Porovnání jednotlivých řešení.

Seznam doporučené odborné literatury:

Roberto Pietro, Luigi V. Mancini *Intrusion Detection Systems (Advances in Information Security)* Springer 2010, ISBN-13: 978-1441945853
Ryan Trost *Practical Intrusion Analysis: Prevention and Detection for the Twenty-First Century: Prevention and Detection for the Twenty-First Century* Addison-Wesley Professional 2009, ISBN-13: 978-0321591807
Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Nevlud**

Datum zadání: 01.09.2013
Datum odevzdání: 29.04.2016

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: 26. Června 2016 

……..……
podpis studenta

Poděkování

Rád bych poděkoval Ing. Pavlu Nevludovi za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Diplomová práce se zabývá možnostmi analýzy anomálií síťového provozu pomocí „open-source“ nástrojů NfSen, Suricata IDS a Bro IDS. Hlavním poznatkem by mělo být, jaké možnosti tyto nástroje poskytují a do jaké míry jsou schopny detekovat tyto anomálie. Jsou zde probrány možnosti uložení dat odchycených na síti pomocí formátů NetFlow a pcap a jejich zpracování ve zmíněných nástrojích. Pro toto ověření bude proveden útok na počítač oběti, který bude ukládat tato data.

Klíčová slova

Analýza anomálií, detekce průniku, bezpečnost, pcap, NetFlow, IDS, IPS, útok, NfSen, Suricata IDS, Bro IDS.

Abstract

The thesis is focused to possibility of network anomaly analysis with open-source tools NfSen, Suricata IDS and Bro IDS. Main goal is to find what opportunities those tools can offer and how deep they are able to detect anomalies. We are going to talk over about how to save captured data with NetFlow and pcap format. Also we will mention how to utilize those data in tools mentioned earlier. To verify those tools, we will make an attack against victim computer.

Key words

Anomaly analysis, Intrusion detection, security, pcap, NetFlow, IDS, IPS, attack, NfSen, Suricata IDS, Bro IDS.

Obsah

Seznam použitých zkratek

Zkratka	Význam
API	Application Programming Interface
BGP	Border Gateway Protocol
BPF	Berkeley Packet Filter
BSD	Berkeley Software Distribution
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Srver
DoS	Denial of Service
ELK	Elasticsearch, Logstash, Kibana
ERSPAN	Encapsulated Remote Switch Port Analyzer
FDDI	Fiber Distributed Data Interface
FIFO	First In First Out
FIN	Typ TCP paketu – Finish
FTP	File Transfer Protocol
Gb	Giga bit
GB	Giga bajt
GMT	Greenwich Mean Time
GNU GPL	General Public License
GPU	Graphic Processing Unit
GRE	Generic Routing Encapsulation
GUI	Graphical User Interface
HDLC	High-level Data Link Control
HIDS	Host Intrusion Detection System
HIPS	Host Intrusion Prevention System
HTTP	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
ICSI	International Computer Science Institute
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IOS	Internetwork Operating System
IP	Internet Protocol
IPFIX	IP Flow Information Export
IPS	Intrusion Prevention System
IRC	Internet Relay Chat
ISO/OSI	International Organization of Standardization – Referenční model ISO/OSI
L2	Layer 2
LAN	Local Area Network
LBNL	Lawrence Berkeley National Laboratory
LTS	Long Term Support
MAC	Media Access Control
MB	Mega Bajt
MD5	Message Digest algorithm
MPL	Multicast Protocol for Low-Power and Lossy Networks
MPLS	Multi Protocol Label Switching
NBA	Network Behavioral Analysis
NIC	Network Interface Card
NIDS	Network IDS

NIPS	Network IPS
NSF	National Science Foundation
PCAP	Packet Capture
PHP	Hypertext Preprocessor
PPP	Point to Point Protocol
PPPoE	PPP over Ethernet
QinQ	IEEE 802.11ad – stacked VLAN
QoS	Quality of Service
RAW	Nezpracovaný soubor
RRD	Round Robin Database
RST	Typ TCP paketu - Reset
SCTP	Stream Control Transmission Protocol
SHA1	Secure Hash Algorithm
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSL	Secure Socket Layer
TB	Tera Bajt
TCP	Transmission Control Protocol
ToS	Type of Service
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VLAN	Virtual LAN
WINPCAP	Windows Packet Capture
WIPS	Wireless IPS

Seznam ilustrací a seznam tabulek

1 Úvod

Mnoho lidí si pod pojmem bezpečnost počítačových sítí představí zabránění průniku útočníka do perimetru sítě tedy prevence, což je jistě velmi důležité. Nicméně pokud už se tak stane je potřeba zjistit, kdo a především jak se naboural do našich systémů. V tom případě přichází na řadu analýza útoku, která samozřejmě může probíhat i kontinuálně. Na základě této analýzy se dozvíme typ proběhlého útoku a mnoho dalších důležitých informací, díky nimž můžeme dalšímu takovému útoku předejít nebo aspoň zmírnit jeho následky. Samostatným odvětvím bezpečnosti je analýza anomálií, jež existuje vícero druhů, nicméně pouze malá hrstka nástrojů dokáže provádět analýzu anomálií na základě chování sítě samostatně.

V této práci postupně rozeberu možnosti analýzy provozu ve sledované síti na základě hlaviček paketů či obsahu celého paketu. Popíšu nástroje pro jejich zpracování a následně provedu útok na sledovaný systém a jeho následnou analýzu.

Druhá kapitola se bude zabývat bezpečností sítí v obecné rovině. Seznámíme se se způsoby její ochrany, využití prostředků pro zabránění průniku do sítě a způsoby analýzy anomálií. Ve třetí kapitole popíšu formát „pcap“, který slouží pro odchycení a následné uložení všech dat z paketů získaných ze sítového provozu. Probereme se jeho historii, ukládáním dat, hlavičky a způsobem oddělení jednotlivých dat v ukládaném souboru. Ve čtvrté kapitole popisuji formát NetFlow, který stejně jako pcap slouží pro záznam provozu, nicméně NetFlow ukládá jen část dat z jednotlivých paketů. Jsou to zdrojová a cílová IP adresa a port, vstupní rozhraní, IP protokol a typ služby. V páté kapitole se zaměřím již na nástroje pro analýzu dat získaných způsoby popsanými v předchozích kapitolách.

Jako jediný zástupce, který zpracovává přímo NetFlow záznamy a má grafické webové rozhraní s mnoha statistikami je NfSen. Druhým nástrojem je Suricata IDS, což je poměrně mladý nástroj, jež je založen na pravidlech, se kterými se porovnává daný provoz a následně při jejich porušení je vygenerován alarm obsahující ono porušení. Třetí a poslední nástroj je již vyzrálý Bro IDS, jehož vývoj započal v polovině devadesátých let. Pro porovnání provozu se zde využívá skriptovacího jazyka, díky nemuž dokážeme podrobně specifikovat, jaké chování na síti očekáváme a při jeho porušení či překročení limitů. Bro IDS vygeneruje záznamy o tomto nežádoucím chování. V poslední šesté kapitole budu provádět samotný útok, který se bude stávat ze dvou samostatných útoků určitém vzorku provozu. Tyto útoky zanalyzujeme postupně ve všech třech nástrojích pro ověření funkčnosti jednoduchosti a náročnosti na systémové prostředky.

2 Analýza síťového provozu

V dnešní době je provoz sítí bez základního monitorovaní provozu téměř nepředstavitelný a je zcela nezbytný. Díky tomuto monitoringu se můžeme dozvědět o případné anomálii v síti, vyznačující se nejčastěji zvýšeným tokem přenášených dat nebo nestandardní hodnotou v paketu. Velmi důležité je, potřeba zjistit, které zařízení či služba může za tuto anomálii.

K tomuto účelu se využívá analýza provozu, a to buď pomocí datových toků jako třeba NetFlow, reprezentující datové toky v síti, které si můžeme představit jako výpis z telefonních hovorů. Vidíme, kdo s kým a jak dlouho telefonoval, ale neznáme obsah komunikace. V počítačové síti tedy sledujeme IP adresy, porty a protokoly komunikujících stran. Z těchto statistik se můžeme dozvědět, že počítač v lokální síti se snaží uložit na server velké množství dat. Nicméně nejsme schopni zjistit jakou mají tato data povahu a jaký byl účel ukládání. Monitorovací nástroj to může považovat za anomálii, jelikož vybočuje ze standardního provozu. Obvykle jsou tyto toky zpracovávány s určitou agregací, tzn. že je zaznamenán například jeden paket z pěti set, což ovšem záleží na velikosti provozu dané sítě.

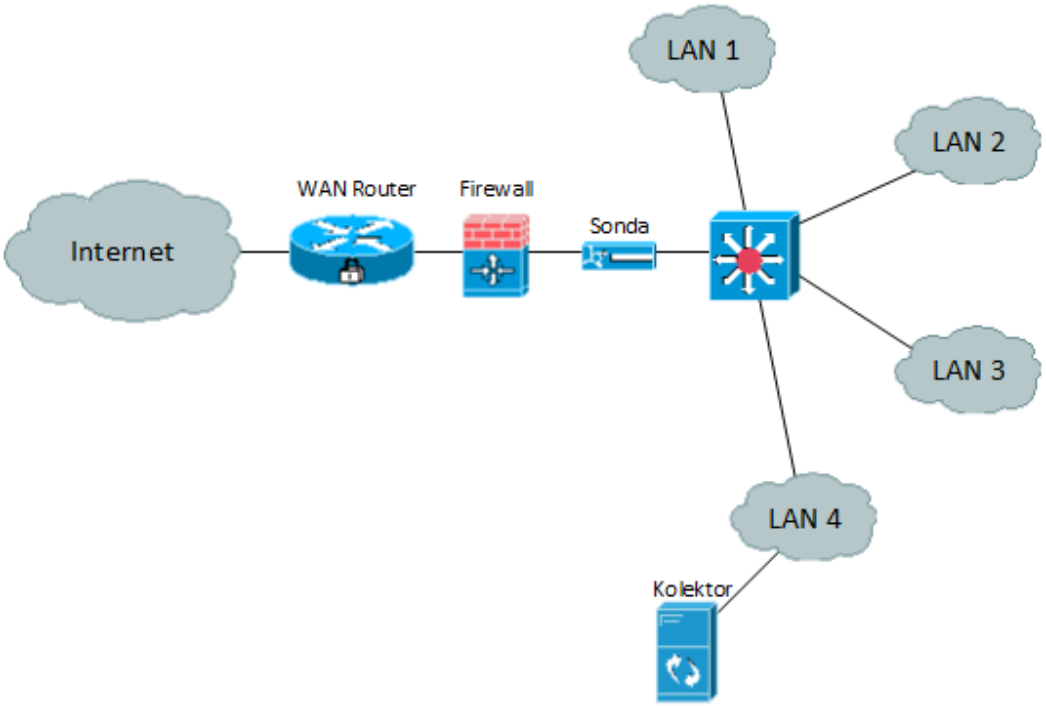
Díky novému standardu IPFIX zavádějící spoustu nových položek do datových toků, jsme schopni získat více informací o dané komunikaci. Tyto nové položky se týkají především aplikační vrstvy. Jedná se o tzv. *„application ID“,* které bylo získáno rozpoznáním aplikace a je přiřazeno k jednotlivým tokům.

Druhým typem monitoringu je paketová analýza, zabývající se analýzou přenášených dat, tedy samotného obsahu komunikace. Tento typ analýzy nepoužívá žádnou agregaci, jelikož bychom přišli o požadovaná data, a tudíž je velmi náročný na diskovou kapacitu, kde je ukládán bez komprese. Nevýhodou je, že bez soukromého klíče nejsme schopni šifrovaný provoz analyzovat, přičemž tohoto provozu v dnešní době rapidně přibývá. Existují dvě možnosti přístupu k této analýze. První typ ukládá veškerý provoz kontinuálně, což je technicky velmi náročné až nepředstavitelné v rychlejších sítích typu 10Gb/s a více, vhodný pouze pro kritické sítě a infrastrukturu. Druhý typ odchytává provoz pouze v případě výskytu nějakého problému na síti, a to zahlcení či ztráta paketu. Nevýhodou tohoto přístupu je, že v případě nějakého incidentu na síti, nemusíme mít jeho záznam a nejsme tedy schopni provést důkladnou analýzu. Správci sítí mají mnoho nástrojů umožňujících odchytávaní paketů, ať už je to *„tcpdump“* pro Linux nebo *„Wireshark“,* jak pro Linux tak pro Windows.

Na základě informací z IPFIX můžeme kombinovat monitoring datových toků s paketovou analýzou spouštěnou na základě těchto toků. Ušetříme tím nároky na výkon hardware a stále budeme mít přehled o spravované síti.

Nejsnadnější způsob jak získat data pro analýzu je nastavení portu na přepínači do tzv. zrcadlení provozu. To znamená, že veškerý provoz, který jím prochází kopíruje na tento port. Může být k němu připojen server pro ukládání dat nebo laptop pro příležitostné vyhodnocování provozu. Druhou možností je vložení tzv. sondy na linku, kterou chceme sledovat. Tato metoda se využívá častěji u kontinuálního ukládání dat, jelikož sonda má vyšší výkon než přepínač a je určena pouze pro tento účel, zatímco přepínač využívá svůj výkon, který je potřeba pro přepínání rámců ke kopírování provozu. Sonda se na připojené lince tváří naprosto transparentně a provoz nijak neovlivňuje.

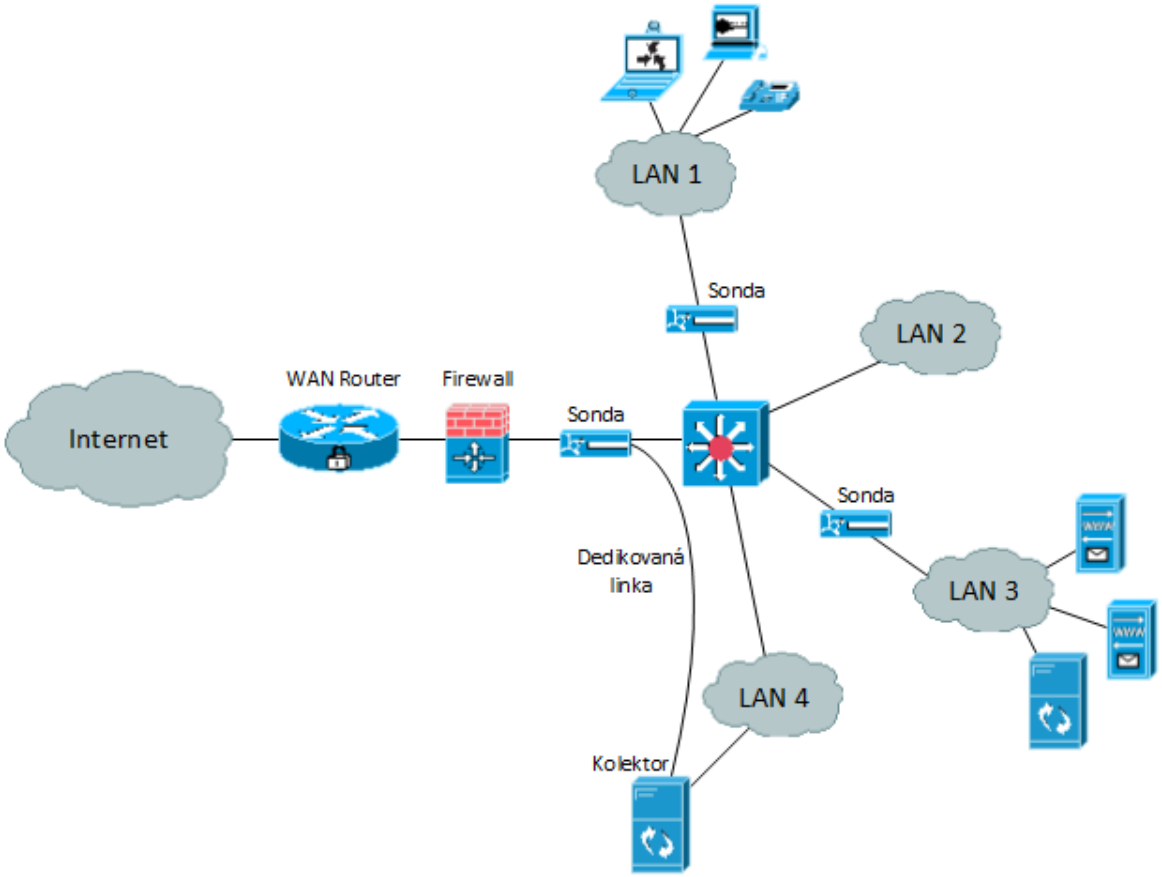
Existuje více možností, kde umístit sondu, tak abychom mohli co nejlépe ochránit naši síť. Sonda může odesílat data jak s datovými toky tak pro paketovou analýzu, záleží pouze na jejím typu a nastavení. První možností je připojení sondy na vstupní lince. Toto řešení vyžaduje velmi výkonou sondu z toho důvodu, že zde prochází veškerý provoz směřující jak do naší LAN sítě tak z ní vystupující. Nevýhodou tohoto řešení je, že neumožňuje detekci hrozeb z vnitřního perimetru sítě. Topologii s touto možností vidíme na obrázku 2.1



Obrázek 2-1: Umístění sondy na hlavní linku

Druhá možnost je umístit sondu na linku do konkretní LAN sítě (např. LAN s velmi důležitými servery), kterou chceme mít nejvíce pod kontrolou. Tato možnost nevyžaduje tak výkonnou sondu, ale máme přehled pouze o této jedné LAN síti. Z toho vyplývá, že nejlepší možností by mohla být kombinace několika sond strategicky rozmístěných. Na hlavní lince bych umístil sondu sledující datové toky a do kritické LAN sítě sondu pro paketovou analýzu. Rovněž je možné vytvořit dedikovanou linku pro odesílaní dat ze sondy na kolektor, jednak aby nevytěžoval datové linky a jednak kvůli bezpečnosti. Pokud by odchycená data byla odesílána skrz běžnou síť (bez jakéhokoliv šifrování či podobné techniky), potencionální útočník by je mohl ovlivnit nebo přinejmenším by viděl, jestli jsou jeho nekalé aktivity zaznamenávány. Topologie takové sítě je na obrázku 2.2

Na obou topologiích vidíme data, jež jsou ukládána na kolektor, což by mohl být server s diskovými poli dostatečné velikosti. K těmto datům je možné přistupovat buď vzdáleně s tím, že nástroj pro analýzu je umístěn jinde v síti nebo je tento nástroj umístěn přímo zde, což není moc vhodné z důvodu již dost velkého vytížení tohoto serveru v případě vetší sítě. Pokud naše síť obsahuje jen několik málo počítačů není problém nástroj umístit přímo na server.



Obrázek 2-2: Rozmístění více sond

2.1 Network IDS/IPS

S rostoucím objemem dat na síti se v hojné míře rozšiřuje automatické vyhodnocování událostí vzniklých na sledované síti. Analýza provozu tedy může být prováděna buď ručně, což je velmi nepraktické ve větších sítích nebo pomocí různých nástrojů, kdy program vyhodnotí možnou událost, kterou poté dále můžeme zkoumat ručně nebo může být provedena automatická akce, ať už spuštění alarmu nebo zablokování nechtěného provozu. Pro tento účel se hojně využívají IDS/IPS systémy. Tyto systémy můžeme popsat jako “bezpečnost = sledování + řízení”. Pro ono sledování využijeme IDS systém a pro řízení využijeme IPS systém.

2.1.1 IDS

IDS je zařízení nebo software pro detekci průniku, který monitoruje síťové nebo systémové aktivity pro odhalení nebezpečných aktivit nebo porušení pravidel. Generuje záznamy o těchto aktivitách, lze tedy zařadit mezi pasivní monitorovací nástroje. Existují dva typy IDS, jedno slouží pro monitorování aktivit v systému samotném (HIDS), druhý typ je pro monitorování aktivit na síti (NIDS). Existují také dva typy bezpečnostního narušení. První tzv. „průnik“, jedná se o útok z vnějšího prostředí organizace a druhý tzv. „zneužití“, což je útok z vnitřní sítě organizace. Systém může být umístěn mimo sledovanou linku a data pro analýzu jsou na něj posílána.

Vlastnosti IDS

- Monitoring a analýza uživatelů a dění v systému,
- analýza systémového nastavení a zranitelností,
- posouzení systémové a souborové integrity,
- rozpoznání útoků dle vzorců,
- analýza anomálií,
- rozpoznání uživatelů porušujících bezpečnostní pravidla.

Základní terminologie použitá v IDS:

- **falešně negativní** – v případě útoku nebyl vygenerován alarm,
- **falešně pozitivní** – vygenerování alarmu, i když neproběhl žádný útok
- **pozitivně negativní** – žádný útok neproběhl a žádný alarm nebyl vygenerován,
- **pozitivně negativní** – v případě útoku byl vygenerován alarm.

NIDS (Systém pro detekci průniku na síti)

Tyto systémy jsou umístěny na strategickém místě nebo na místech pro monitoring provozu do a ze zařízení umístěných v síti. Provádí analýzu přenášených dat, které porovnává s knihovnou známých útoku či s pravidly, která jsme definovali. Existují dva typy NIDS Jeden (on-line) se využívá pro analýzu v reálném čase a druhý (off-line) slouží pro zpětnou analýzu již uložených dat.

V porovnání s firewallem IDS pouze generuje alarm v případě porušení pravidel, naopak firewall pouze zastaví nežádoucí provoz, ale nikoho o tom neinformuje. Pokud bychom chtěli tyto systémy spojit musíme použít IPS, což je IDS s funkcí zahození nebo odmítnutí nežádoucího provozu.

2.1.2 IPS

IPS neboli „Systém prevence průniku“, je systém pro monitorování sítě a stejně jako IDS generuje alarmy s tím, že může zablokovat nežádoucí provoz. Stejně jako IDS provoz se porovnává vytvořenými pravidly nebo dle naučené „normálnosti“ sítě. Na rozdíl od IDS je však tento systém umístěn přímo na sledované lince, aby onen nežádoucí provoz mohl zastavit.

Typy IPS

- **NIPS** – monitoruje síť, aby blokoval nežádoucí provoz,
- **WIPS** – monitoruje bezdrátové sítě, aby blokoval nežádoucí provoz,
- **NBA** – monitoruje síť a dělá si vzor sítě pro zjištění neobvyklého provozu,
- **HIPS** – monitoruje konkrétní PC a analyzuje software a události probíhající na tomto PC.

Vlastnosti IPS

- Generování alarmů a odesílaní administrátorovi,
- zahazování nebezpečných paketů,
- zablokování provozu z konkrétní IP adresy,
- resetování spojení.

Metody detekce

- **Porovnávání signatur**

Procházející provoz je porovnáván se signaturami neboli pravidly, která jsme vytvořili. Na základě tohoto porovnání je provoz buď zahozen a vygenerován alarm, propuštěn a vygenerován alarm nebo je propuštěn tak, jak je specifikováno v jednotlivých pravidlech. Pravidla jsou psána jednoduchou formou pro porovnávání na úrovni paketů.

- **Statické anomálie**

Můžeme vytvořit sadu pravidel, která jsou založena na znalosti protokolů, a kterými specifikujeme „normální“ provoz. V případě, že provoz vybočuje z mezí těchto pravidel, je s ním dále nakládáno dle specifikace v pravidlech. Způsob se velmi podobá „porovnávání signatur“ ,naproti nim ale nabízí pokročilejší možnosti na úrovní aplikačních protokolů.

- **Behaviorální analýza**

Metoda závisí na vytvoření vzoru provozu, kde předpokládáme, že tento je „normální“. Pro vytvoření tohoto vzoru je potřeba delší čas, aby byl co nejpřesnější. Pokud je na síť veden útok a my o něm nevíme a začneme vytvářet vzor, je zcela jisté, že další podobný útok nedetekujeme, jelikož bude považován za „normální“. Za předpokladu, že máme „normální“ vzor, systém nás při jakémkoli vybočení z daných mezí informuje o nepředvídané aktivitě. [1]

3 Zachycení a rozbor dat v souboru *.pcap

3.1 Co je to *.pcap?

Sítí prochází mnoho dat, která mohou být monitorována, zachycena a dále zpracována, ať už útočníkem z důvodu sledování a analýzy hesel, tajných údajů atd., nebo bezpečnostním expertem z důvodu zabezpečení sítě. Zpracovaní dat bez zásahu do soukromí (bezpečnostním expertem) je použití pouze hlaviček z těchto odchycených paketů. Jeden z nejpoužívanějších formátů, ve kterém se tyto hlavičky paketů zpracovávají, je ”pcap”. Pcap poskytuje filtrovací funkci pro odchytávání paketů, jenž využívá mnoho nástrojů pro monitorování sítě jako jsou systémy detekce průniku, generátory provozu atd. Například ”wireshark”, ”tcpdump“ a mnoho dalších, které jsou sofistikovanější. Data zde můžeme vidět v reálném čase jak prochází sítí a dále je můžeme ukládat, filtrovat nebo znova načíst pro zpětnou analýzu. Pro podporu v systémech UN*X se využívá knihovna “libpcap” a pro uživatele WINDOWS je to knihovna “winpcap”. Obě knihovny využívají stejný formát a jsou vzájemně kompatibilní, což znamená, že data odchycená v systému UN*X můžeme znova zobrazit v systému WINDOWS a naopak. Libpcap byl vyvinut výzkumnou skupinou “tcpdump“ při Lawrence Berkeley Laboratory, která má na svědomí mnoho dalších síťových nástrojů. Knihovna libpcap je napsaná v programovacím jazyce C. S využitím této knihovny můžeme napsat svůj vlastni program pro odchytávání paketů. Pcap formát podporuje různé typy médií, mezi hlavní patří samozřejmě ethernet. Podpora médií záleží na použitém operačním systému. Například Bluetooth lze odchytávat pouze na systému linux.

3.2 Zachycení dat a uložení do souboru *.pcap

Jako příklad pro zachycení dat jsem zvolil paketový analyzátor wireshark, který je šířen pod licencí GNU GPL a pro své fungování potřebuje knihovnu libpcap. Po nainstalování a spuštění tohoto analyzátoru jsme vyzvání k volbě fyzického rozhraní (NIC), na kterém chceme odchytávat pakety a spustit odchytávání. V základním nastavení se otevřou tři okna nad sebou, v horním se zobrazují odchycené pakety, v prostředním okně hlavičky vybraného paketu a ve spodním okně „hexa“ kód vybrané hlavičky. Odchytávání necháme spuštěné tak dlouho, kolik dat nebo jaká chceme odchytávat. Po ukončení zachytávání, zůstanou pakety zobrazeny a můžeme s nimi dále pracovat, filtrovat nebo je ukládat.

3.3 Analýza souboru *.pcap (formát)

Formát souboru pcap se téměř nezměnil od roku 1998, kdy byl ve verzi 0.4. Jediný rozdíl je ve formátu pcapng, viz níže.

Od verze 1.5.0 pcap podporuje záznam paketů v nanosekundovém intervalu. Soubor se skládá z globální hlavičky, která je v souboru pouze jednou a obsahuje globální informace následované nulou, nebo několika dalšími záznamy pro každý odchycený paket.

*Tabulka 2.1: Formát souboru *.pcap*

Globální hlavička	Hlavička paketu 1	Data paketu 1	Hlavička paketu 2	Data paketu 2	...

Tabulka 2.2: Formát Globální hlavičky

Kouzelné číslo	
Hlavní verze	Vedlejší verze
Korekce vůči GMT	
Upřesnění času	
Velikost paketů	
L2 rámec	

Globální hlavička má fixní velikost 24 bajtů:

- 4 bajty – kouzelné číslo „magic number“,
- 4 bajty – hlavní verze „ major version“ 2 bajty a vedlejší verze „minor version“ 2 bajty,
- 4 bajty – korekce času vůči GMT v sekundách,
- 4 bajty – přesnost časové značky odchyceného paketu,
- 4 bajty – velikost odchycených paketů,
- 4 bajty – typ rámce 2. vrstvy modelu ISO/OSI (Ethernet, PPP, HDLC, atd.).

Globální hlavičku si můžeme zobrazit v hexadecimálním formátu pomocí příkazu *„hexdump“*.

probe@probe:~$ hexdump -n 24 -C rarp_request.cap

d4 c3 b2 a1 02 00 04 00 00 00 00 00 00 00 00 00 ff ff 00 00 01 00 00 00

První 4 bajty **0xd4c3b2a1** představují kouzelné číslo, které je použito pro identifikaci souboru, jedná o zápis pomocí little endianess. Pokud by hodnota byla ve formátu **0xa1b2c3d4** (big endianess), znamená to, že ostatní položky v hlavičce souboru a hlavičky jednotlivých paketů jsou zapsány ve stejném pořadí bajtů tak, jak je počítač zpracovává. V opačném případě se musí tyto bajty uspořádat do správného formátu. Použitý zápis můžeme zjistit pomocí příkazu “*file”*

probe@probe:~$ file rarp_request.pcap

rarp_request.pcap: tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length 262144)

Další 4 bajty **02 00 04 00** jsou hlavní a vedlejší verze, v našem případě 2.4. Důvodem proč se hlavní verze zapisuje jako 0x0200 a ne 0x0002, je pokračující trend formátu little endianess. Následující 4 bajty **00 00 00 00** jsou prezentovány jako úprava časové zóny v hlavičce oproti GMT,

v našem časovém pásmu by to bylo v případě zimního času -3600. Další 4 bajty **00 00 00 00** představují přesnost časové značky a jsou vždy nastaveny na 0. Následují 4 bajty **ff ff 00 00** indikující maximální délku jednotlivých odchycených paketů v bajtech. Odchycený paket v daném odchytávaném souboru nemusí nutně obsahovat všechna data, jež jsou obsažena v paketu, který je prezentován na síti. Odchytávaný soubor může obsahovat nanejvýš prvních N bajtů každého paketu. Hodnota N, tzv. "snapshot length" nebo "snaplen", může být větší než největší možná velikost paketu, obvyklá hodnota pro tcpdump a wireshark je 65535 bajtů, což je i v našem případě. Poslední 4 bajty **01 00 00 00** identifikují typ rámce 2. vrstvy OSI/ISO modelu, v tomto případě to je IEEE 802.3-ethernet.

Hlavička paketu má velikost 16 bajtů:

- 8 bajtů – časové razítko,
- 4 bajty – velikost dat odchyceného paketu v *.pcap souboru. Tato velikost by nikdy neměla, přesáhnout velikost určenou v globální hlavičce.
- 4 bajty – velikost odchyceného paketu jak byl přenášen po síti.

probe@probe:~$ hexdump -C rarp_request.pcap -s 24 -n 16 | cut -c 11-59

3a ac 62 56 25 d0 0b 00 ea 05 00 00 ea 05 00 00

První 4 bajty **3a ac 62 56** jsou časové razítko v sekundách od 1. ledna 1970 00:00:00 GMT, v systému *Unix je můžeme převést na reálný časový údaj. Všimněte si opačného pořadí zadaného údaje pro správný převod.

probe@probe:~$ calc 0x5662ac3a

1449307194

probe@probe:~$ date --date='1970-01-01 1449307194 sec GMT'

So pro 5 10:19:54 CET 2015

Stejné to je pro následující 4 bajty, zpřesňují časové razitko v mikrosekundách.

probe@probe:~$ calc 0x000bd025

774181

Třetí pole 4 bajtů **ea 05 00 00** představuje velikost odchyceného paketu, což je standardní velikost ethernetového paketu tj. MTU 1514 obsahující i hlavičku. Poslední 4 bajty **ea 05 00 00** jsou stejné a zobrazují velikost paketu jak byl reálně interpretován na síti. Tyto hodnoty se liší od maximální velikosti udávané v globální hlavičce, která představuje pouze maximální možnou velikost celého odchyceného souboru. Dále už následují data. [2][3][4][5][6]

3.3.1 **Pcapng**

PCAP Next Generation Dump File Format, nebo zkráceně pcapng je rozšíření pro stávající a stále velmi používaný pcap. Pcapng se skládá z bloků, kde některé jako „hlavička sekce" jsou povinné a některé jsou nepovinné Celek může být složen z několika „hlaviček sekce" a k nim přiřazených dat. Díky nepovinným blokům skýtá velké možnosti rozšíření. Rozšíření fungují na podobném principu jako u IPv6. [7]

3.4 Rozbor dat v souboru *.pcap

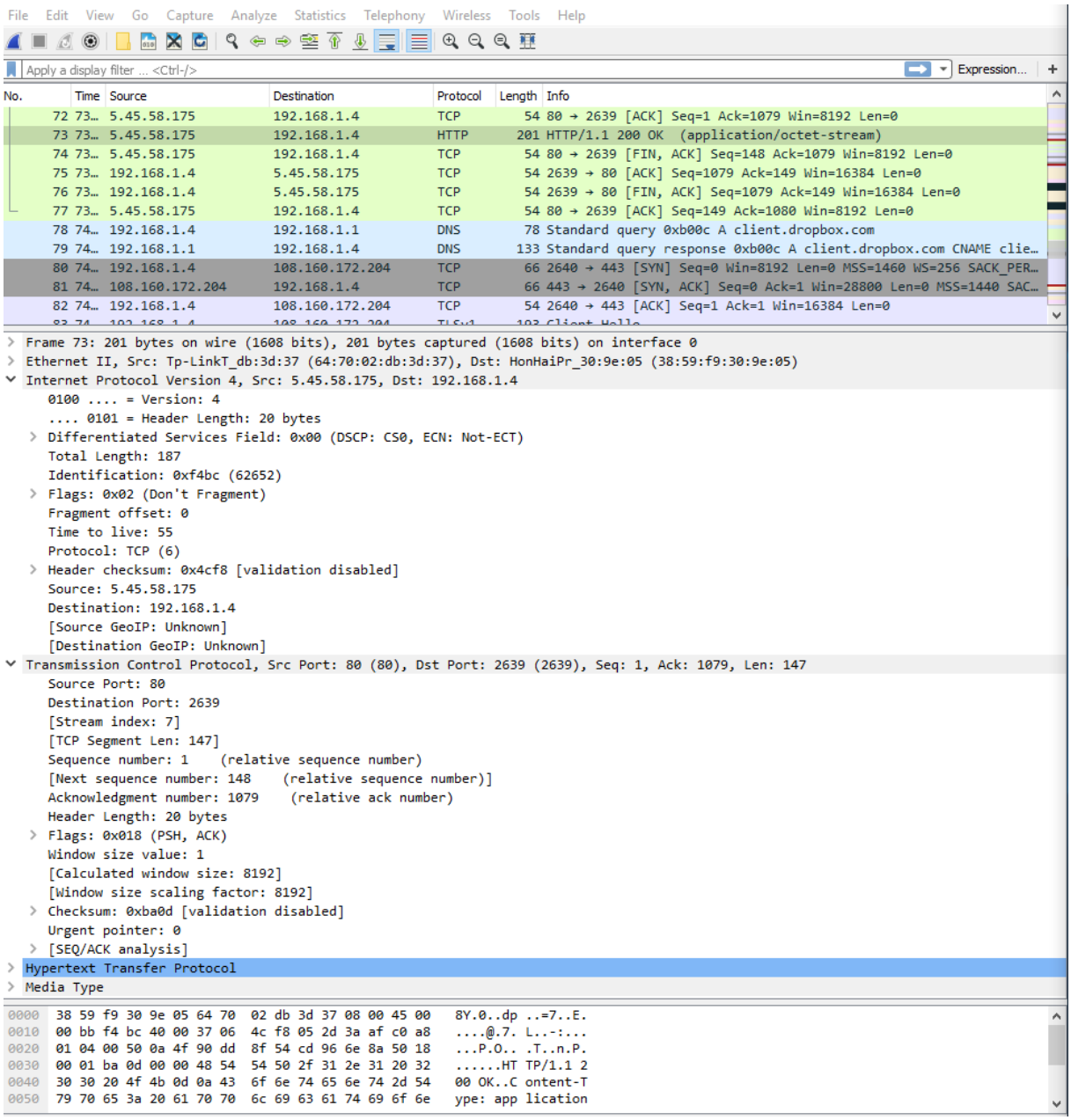
3.4.1 Wireshark

Wireshark je jeden z nejznámějších open-source nástrojů s grafickým uživatelským rozhraním, šířený pod licencí GPL pro analýzu paketů. Je napsán v jazyce C, C++ a jeho původní verze vyšla kolem roku 1988. Je porovnatelný s placenými aplikacemi a svou funkčností je velmi podobný nástroji *tcpdump,* běžícímu v příkazové řádce Linux systémů. Tento nástroj se snaží odchytávat pakety na síti a zobrazit je co možná nejdetailněji. Wireshark je určen pro všechny, jež potřebuji vědět co se děje na jejich síti od administrátora sítě přes vývojáře, kteří potřebují zjistit jak se jejich program chová na síti, až po všechny ostatní, kteří jsou jen zvědaví. Zachycená data mohou být uložena do souboru pro pozdější analýzu ve formátu pcap.

Vlastnosti:

- dostupné jak pro Linux tak pro Windows,
- odchycení dat aktuálně tekoucí sítí,
- ukládání do otevřeného formátu pcap, pro možnost otevření v dalších aplikacích,
- velmi detailní zobrazení obsahu paketů,
- filtrování/hledání paketů podle velkého množství kriterií,
- vytváření statistik,
- složení VoIP hovoru z jednotlivých paketů,
- rozdělení protokolů pomocí barev,
- a mnoho dalších.

Použití nástroje Wireshark je velmi intuitivní. Po spuštění vás přivítá úvodní obrazovka s možností volby portu, na kterém chceme sledovat přenášená data. Po výběru portu Wireshark začne okamžitě odchytávat data. V horní části se nachází pole pro vkládání filtrů pro zobrazení pouze chtěných protokolů či komunikujících stran. Pokud toto pole zůstane prázdné a za předpokladu, že naše síťová karta je nastavena do promiscuous módu (zobrazí i data, která nejsou přímo určena nám) uvidíme všechna protékající data v dalším bloku, který se automaticky posouvá s přibývajícími daty. V podstatě je to pouze seznam všech odchycených paketů. V dalším bloku je detailní pohled na vybraný paket, vidíme zde data od fyzické vrstvy až po aplikační s ohledem na vybraný paket. Poslední blok obsahuje surová data v hexa formátu tak, jak jsou přenášena po médiu. Celý tento pohled na základní strukturu nástroje můžeme vidět na obrázku 3.1. [8]



Obrázek 3-1: Pohled na nástroj Wireshark

3.4.2 Rozbor DHCP komunikace

Jako ukázku jsem zvolil odchycení paketů požadavku klienta na IP adresu z DHCP serveru. Požadavek se skládá ze 4 paketů, které můžeme vidět na obrázku 3.2. Source = zdrojová IP adresa ze které byl paket odeslán, Destination = cílová IP adresa na kterou byl paket odeslán, Protocol = protokol pomocí kterého byl paket přenesen sítí, Lenght = velikost paketu v bajtech tak, jak je prezentován na fyzické vrstvě, Info = doplňující informace o typu zprávy, případně sekvenční čísla, potvrzovací čísla, velikost okna atd. Rozbor jednotlivých paketů je popsán níže.

Rozbor a ukázky jsou provedeny v aplikaci Wireshark.

Source	Destination	Protocol	Length	Info
0.0.0.0	255.255.255.255	DHCP	618	DHCP Discover - Transaction ID 0x155c
192.168.0.1	255.255.255.255	DHCP	342	DHCP Offer - Transaction ID 0x155c
0.0.0.0	255.255.255.255	DHCP	618	DHCP Request - Transaction ID 0x155c
192.168.0.1	255.255.255.255	DHCP	342	DHCP ACK - Transaction ID 0x155c

Obrázek 3-2: DHCP

3.4.3 DHCP Discover

Každý rámec v případě Ethernetu obsahuje cílovou a zdrojovou MAC adresu, která následuje ihned po velikosti odchyceného paketu. Jak můžeme vidět na obrázku 3.3. První řádek určuje velikost odchyceného paketu a typ druhé vrstvy OSI modelu. Druhý řádek obsahuje MAC adresy, v případě DHCP Discovery paketu je zdrojová MAC adresa zařízení, které odeslalo tento požadavek. Cílová adresa je typu broadcast, jelikož zařízení nemá žádné informace o síti, je požadavek rozeslán na všechny zařízení v broadcast doméně. V třetím řádku jsou obsaženy IP adresy, jelikož zařízení žádnou IP nemá, je jako zdrojová adresa zvolena 0.0.0.0 a cílová IP adresa je broadcast. Dle specifikace klient komunikuje na portu 68 a DHCP server naslouchá na portu 68. Bootstrap Protokol obsahuje typ požadavku, identifikaci, jméno klienta a další informace.

```
Frame 1: 618 bytes on wire (4944 bits), 618 bytes captured (4944 bits)
Ethernet II, Src: cc:00:0a:c4:00:00 (cc:00:0a:c4:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
User Datagram Protocol, Src Port: 68 (68), Dst Port: 67 (67)
Bootstrap Protocol (Discover)
```

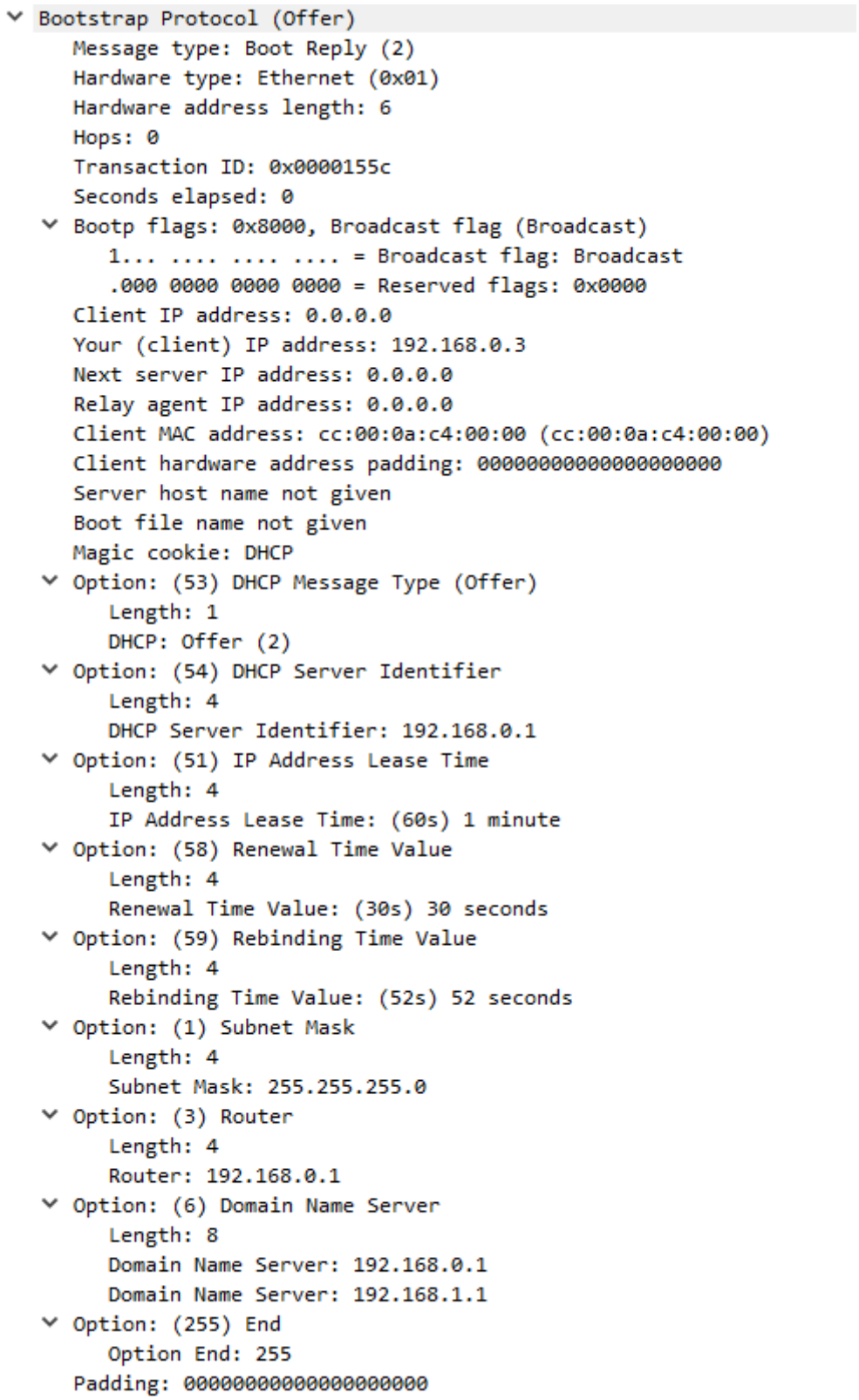
Obrázek 3-3: DHCP Discover

3.4.4 DHCP Offer

Tímto paketem DHCP server odpovídá na požadavek klienta. Na obrázku 3.4 vidíme jako zdrojovou MAC adresu tuto DHCP adresu serveru a jako cílová MAC adresa je opět použit broadcast. Jelikož server již má svou IP adresu, je tato použita v paketu jako zdrojová, ale cílová IP adresa je opět broadcast. Porty jsou použity stejné, pro server 67 a pro klienta 68. Bootstrap Protocol obsahuje v tomto případě mnohem více informaci, a to nabízenou IP adresu, masku a výchozí bránu, stávající IP adresu klienta, což je stále 0.0.0.0, MAC adresu klienta, identifikátor a IP adresu DHCP serveru, IP adresu routeru, který je v tomto případě shodný s DHCP serverem, propůjční dobu a další, jak můžeme vidět na rozšířeném pohledu na Bootstrap protocol na obrázku 3.5.

```
Frame 2: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
Ethernet II, Src: cc:01:0a:c4:00:00 (cc:01:0a:c4:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 255.255.255.255
User Datagram Protocol, Src Port: 67 (67), Dst Port: 68 (68)
Bootstrap Protocol (Offer)
```

Obrázek 3-4: DHCP Offer



Obrázek 3-5: Bootstrap Prtocol (Offer)

3.4.5 DHCP request

Tento paket se liší oproti DHCP Discovery pouze v poslední položce, kde klient požádá DHCP server o nabízenou IP adresu. Klient stále nemůže využít nabízenou IP adresu, tudíž je jako zdrojová IP použita 0.0.0.0.

```
Frame 3: 618 bytes on wire (4944 bits), 618 bytes captured (4944 bits)
Ethernet II, Src: cc:00:0a:c4:00:00 (cc:00:0a:c4:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
User Datagram Protocol, Src Port: 68 (68), Dst Port: 67 (67)
Bootstrap Protocol (Request)
```

Obrázek 3-6: DHCP request

3.4.6 DHCP ACK

Tímto paketem DHCP server potvrzuje vypůjčenou IP adresu, odešle ji na broadcast adresu tak aby o ní věděli všichni klienti v broadcast doméně.

```
Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
Ethernet II, Src: cc:01:0a:c4:00:00 (cc:01:0a:c4:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 255.255.255.255
User Datagram Protocol, Src Port: 67 (67), Dst Port: 68 (68)
Bootstrap Protocol (ACK)
```

Obrázek 3-7: DHCP ACK

4 Popis protokolu NetFlow

NetFlow je otevřený protokol, který vyvinula společnost Cisco Systems pro monitorování provozu na síti. Jakožto vestavěný nástroj v Cisco IOS jej stačí zapnout pomocí příkazu a specifikovat vstupní nebo výstupní provoz na daném portu. Analyzováním NetFlow dat může administrátor zjistit například IP adresy, které na síti generují nejvíc provozu a kudy tento provoz teče.

4.1 Použití NetFlow

- Monitorování sítě: monitorování dat téměř v reálném čase. Analýza může být použita k vizualizaci provozu z jednotlivých směrovačů či přepínačů a pro-aktivní detekci problému.
- Monitorování aplikací a profilování: umožňuje administrátorovi získat detailní přehled aplikací, které nejvíce komunikují na síti. Tyto informace pomohou lépe naplánovat design sítě a alokovat zdroje.
- Monitorování uživatelů: získání detailního přehledu o využití sítě uživateli. Stejně jako u monitorování aplikací slouží k lepší alokaci zdrojů a detekci potenciálního narušení bezpečnosti a pravidel.
- Plánování sítě: díky dlouhodobému odchytávání dat lze plánovat růst sítě, zvyšovat počet portů, šířku pásma případně vylepšit směrování v síti. NetFlow detekuje nežádoucí provoz, analyzuje nově zjištěné aplikace na síti a dává nám hodnotné informace pro snížení nákladů vynaložených pro správu sítě.
- Bezpečnostní analýza: identifikace a klasifikace DDoS útoků, virů a červů v reálném čase. Změny v síťovém chování, které indikují anomálie a můžou být původem útoku. Data se dají použít ke zpětné analýze proběhnutého útoku.
- Účtování/fakturace: Poskytovatel služeb je schopen pomocí NetFlow účtovat na základě času, použité šířky pásma, QoS nebo použití aplikací.
- Ukládání a vytěžování dat: NetFlow data se ukládají lokálně do mezipaměti, a poté na vzdálený server. Data mohou být později použita pro zjištění “kdo”, ”co”, ”kde” a ”jak dlouho”.

NetFlow se skládá ze dvou klíčových komponent: serveru (tzv. kolektroru) pro uložení dat a transportního mechanismu, který data posílá. Aktuální verze NetFlow je č. 9 a je také jako standard IETF s názvem IPFIX (IP Information eXport). Hlavní zaměření NetFlow bylo vždy na tok IP dat, to se ale s příchodem verze č. 9 změnilo a nyní také obsahuje informace o L2, IPv6, Multicastu, MPLS, BGP a další.

4.2 Složení NetFlow

Tok dat je definován jako jednosměrný proud paketů mezi zdroje a cíle s danými IP adresami a porty. V NetFlow je tok definován jako kombinace následujících:

- zdrojová IP adresa ,
- cílová IP adresa ,
- zdrojový port,
- cílový port,
- protokol 3. vrstvy OSI/ISO modelu,
- ToS,
- rozhraní, na kterém byl tok odchycen.

Těchto sedm hodnot definuje unikátní tok. Pokud se liší jedna jediná hodnota, jedná se již o další tok. Tok obsahuje ještě další pole, která se liší podle verze NetFlow.

4.3 Vytvoření záznamu v mezipaměti a export dat

Mezipamět obsahuje NetFlow záznamy o všech aktivních tocích. Záznam je vytvořen při detekci první paketu nového toku se stejnými hodnotami identifikujícími stejný tok, který prochází síťovým zařízením. Záznamy NetFlow jsou exportovány na NetFlow server periodicky na základě nastaveného času. Velikost exportovaných dat je cca. 1.5% velikosti všech dat, které projdou sledovaným síťovým zařízením. Tato velikost odpovídá datům, která nejsou vzorkovaná.

4.3.1 Expirace záznamů v paměti

- Toky, které jsou nečinné po předem definovanou dobu jsou odebrány z mezipaměti.
- Toky, které jsou příliš dlouhé a překročí maximální délku 30 minut jsou odebrány z mezipaměti.
- TCP spojení, které je ukončené, tzn. obsahuje FIN paket nebo RST paket, expiruje.
- Pokud je mezipamět plná, aplikuje se agresivnější přístup k odebírání dat z mezipaměti.

Toky, které expirují jsou odesílány na server v tzv. “NetFlow Export datagramech”. Tyto datagramy se skládají až ze 30-ti záznamů ve verzi 5 nebo 9. NetFlow zprávy jsou děleny podle rozhraní, na které byly přijaty. Aby mohla být data exportována, je potřeba nakonfigurovat IP adresu a port serveru tzv. NetFlow kolektoru.

4.4 Formát NetFlow

NetFlow export datagram se skládá z NetFlow hlavičky a sekvence NetFlow toků. Hodnoty v hlavičce se mohou lišit dle aktuální verze. Jednotlivé toky obsahují IP adresy, porty atd. Ukázka typického NetFlow datagramu je v tabulce.

Tabulka 3.1: NetFlow datagram

IP hlavička
UDP hlavička
NetFlow hlavička
NetFlow tok 1
…
NetFlow tok n

4.5 Verze NetFlow

Verze 1 byla původní verzí, jež byla podporovaná zařízeními Cisco a zřídka je použita i dnes. Verze 2, 3, 4, byly pouze interní verze, a nikdy nebyly vydány. Verze 6 již není podporována. Ve verzi 5 byla přidána podpora BGP. Verze 7 je podporována pouze na přepínačích řady Cisco Catalyst. Verze 8 je téměř stejná jako verze 5.

4.5.1 Verze 1

Původní verze, kterou Cisco vydalo. Byla podporována pouze na jejich zařízeních a zřídka se používá dodnes. Hlavička má velikost 17 bajtů a skládá z:

- verze,
- počtu exportovaných NetFlow toků (maximálně však 24),
- času, jak dlouho je zařízení zapnuto,
- unix času v sekundách a
- unix času v nanosekundách.

NetFlow tok má velikost 49 bajtů a obsahuje data o jednotlivých tocích, kterých může být v jednom datagramu maximálně 24.

4.5.2 Verze 5

Nejpoužívanější verze NetFlow, která byla vydána roku 2009. Podporuje pouze IPv4 a v jednom datagramu 30 toků. V dnešní době byla ve většině případů nahrazena novější verzí 9.

4.5.3 Verze 9

Tato verze prošla největšími změnami napříč všemi verzemi. IETF vydala vlastní verzi (IPFIX), která je založená verzi 9.

Ze směrovače nebo přepínače může být exportována téměř jakákoliv informace ,jako jsou informace vrstev 2 až 7, směrovací informace, IPv4, IPv6, multicast, MPLS.

Hlavička NetFlow verze 9 je složena z:

- verze,
- počtu exportovaných NetFlow záznamů,
- času, jak dlouho je zařízení zapnuto,
- unix času v sekundách,
- sekvenčního čísla, které identifikuje jednotlivé exportované pakety pro případ jejich ztráty při přenosu,
- ID zdroje, což je 32 bitová hodnota, jednoznačně identifikující zdroj exportovaných dat. Kolektor používá tuto hodnotu pro jednoznačnou identifikaci zdroje NetFlow datagramu společně s IP adresou zdroje.

Tento formát NetFlow je dnes nejrozšířenější verzí, proto jej zde popíši podrobněji.

NetFlow v9 se výrazně liší od všech svých předchůdců. Hlavní rozlišující vlastnost je ta, že tento formát je založen na šablonách. Šablony poskytují lepší možnost úpravy exportovaných toků dle potřeby a především pro možné pozdější rozšíření či použití.

Hlavními výhodami použití šablon jsou:

- Partneři třetích stran, vyvíjející aplikace, ať už v roli kolektoru, exportéru nebo aplikace pro zobrazení a analýzu NetFlow dat. V případě přidání nových vlastností v NetFlow nemusí vydávat novou verzi této aplikace, ale stačí pouze změnit externí soubor popisující tento nový formát šablon.
- Nové vlastnosti jsou přidávány rychleji, bez omezení provozu.
- NetFlow je připravené na budoucí rozšiřování díky adaptabilitě verze 9.

Části (pojmy) popisující NetFlow v9:

- Exportovaný paket - vytvořený zařízením (router,, switch, atd.) s povolenou službou NetFlow, dále je pak tento paket odesílán na kolektor, kde je následně zpracován.
- Hlavička paketu - první část exportovaného paketu, poskytuje základní informace o odesílaném paketu jako je verze NetFlow, počet záznamů v paketu, sekvenční číslo, detekující ztracený paket.
- Flowset - následuje po hlavičce paketu. Paket obsahuje informace, které jsou analyzovány a interpretovány na kolektroru. Flowset je obecný termín pro sbírku záznamů v exportovaném paketu, Existují dva druhy Flowset, vzor a data. Exportovaný paket obsahuje jeden nebo více Flowset záznamů a oba typy mohou být prezentovány v jednom exportovaném paketu.
- Flowset vzor - je sbírka jednoho nebo více vzorových záznamů, které byly seskupeny do jednoho paketu.
- Vzorový záznam - definuje formát následujících dat, která mohou být přijata v aktuálním nebo následujím exportovaném paketu. Vzorový záznam nemusí nezbytně definovat formát

dat ve stejném paketu. Kolektor proto musí udržovat v mezipaměti všechny vzorové záznamy, které přijal a dle nich analyzovat přijatá data.

- ID vzoru - je unikátní číslo, rozlišující jednotlivé vzorové záznamy z jednoho exportního zařízení. Tato unikátnost nemusí být zaručena v případě použití více exportních zařízení. Kolektor by měl z tohoto důvodu také ukládat do mezipaměti IP adresu exportního zařízení pro zaručení této unikátnosti.
- Data Flowset - je sbírka datových záznamů, které byly seskupeny v jednom exportním paketu.
- Datové záznamy - obsahuje jednotlivé IP toky, které jsou prezentovány na exportním zařízení. Každá skupina datových záznamů se vztahuje k již dříve zmíněnému ID vzoru
- Vzorové nastavení - speciální typ vzorového záznamu, do kterého se zaznamenává formát dat z NetFlow procesu.
- Nastavení datového záznamu - speciální typ datového záznamu (založený na vzorovém nastavení) s rezervovaným ID záznamem, který poskytuje informace o NetFlow procesu samotném.

Záznam NetFlow v9 se skládá z hlavičky paketu a nejméně jednoho vzorového data nebo flowsetu.

4.6 Výkon Zařízení

Sběr dat ve směrovačích a přepínačích ovlivňuje výkon těchto zařízení. Využití procesoru závisí na počtu udržovaných toků v mezipaměti. Při 10000 aktivních tocích je využití CPU o 4% vyšší, při 65000 aktivních tocích je využití CPU vyšší dokonce o 16%. Zátěž procesoru lze snížit pomocí vzorkování odchytávaných paketů. Vzorkováním NetFlow v poměru 1:1000 dokáže snížit tuto zvýšenou zátěž až o 82% a 1:100 ji sníží o 75%. [9][10][11]

5 Nástroje použité pro analýzu provozu

5.1 NfSen

NfSen je graficky a webově založená nadstavba pro zobrazení dat zpracovávaných nástrojem NfDump, díky němuž můžeme sledovat vývoj událostí či incidentů v naší síti snadno pomocí grafů. Ve webovém prostředí lze přizpůsobit filtry jak pro sledované protokoly 3. a 4. vrstvy, tak pro zobrazení určité délky časové osy. Tyto filtry můžeme nastavit pro jednotlivé profily a mezi nimi pak jednoduše přepínat abychom nemuseli pokaždé filtry nastavovat znova. Dále lze přepínat mezi zobrazením grafu na základě toků či paketů, můžeme zobrazit historii dění na síti a porovnávat se současnou situací. NfSen dovoluje nastavit mezní hodnoty, při kterých nás upozorní, že byly překročeny pomocí emailu nebo ve webovém prostředí. Důležitou součástí NfSen pro vykreslování grafů je databáze RRD. NfSen je napsán v PHP (webové rozhraní) a Perl, a měl by se dát spustit na jakémkoli stroji se systémem linux.

NfSen umožňuje rozšíření pomocí pluginů, které si můžeme napsat sami nebo použít již vytvořené. Mezi vytvořené patří:

- SURFmap – lokalizace IP na základě Google Maps API,
- SSHCure – SSH IDS,
- Nfsight – vizualizace komunikace klient/server,
- PortTracker – reporty na základě portu,
- Botnet – porovnání útoku se známým seznamem botnetů.

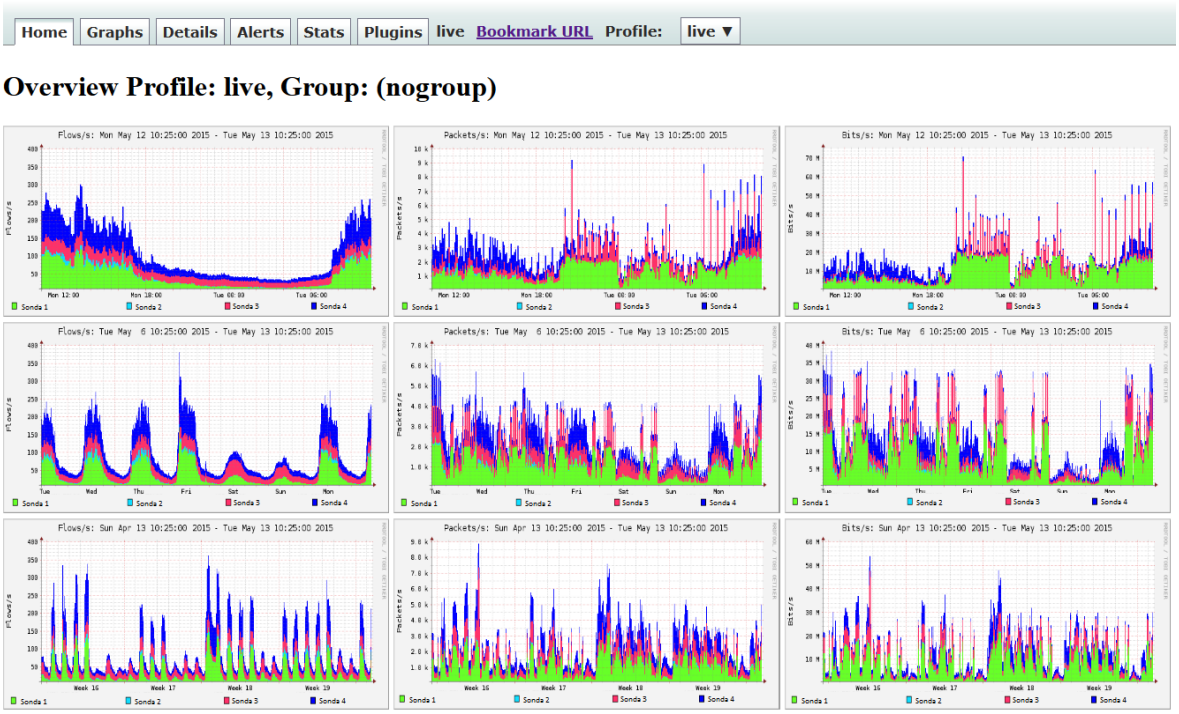
5.1.1 Instalace NfSen

Instalaci nástroje NfSen jsem provedl v operačním systému Ubuntu 12.04.5 LTS (GNU/Linux 3.13.0-83-generic x86_64). Byla provedena aktualizace repositářů a následně i systému.

Pomocí příkazu *„apt-get install“* jsem nainstaloval služby potřebné pro běh NfSen, což jsou gcc, apache2, libapache2-mod-php5, php5-common, libmailtools-perl, rrdtool, librrds-perl, libio-socket-ssl-perl. Dále je potřeba nainstalovat modul pro perl pomocí příkazu „perl -MCPAN -e shell“. Nástroj NfSen jsem stáhl z oficiálních stránek projektu. Před první instalací jsem upravil konfigurační soubor (soubor se nachází v */etc/nfsen.conf*), který může být upravován i následně. Nástroj jsem nainstaloval pomocí příkazu „*cesta k NfSen/install.pl /etc/nfsen.conf*“, kde *„install.pl“* je skript pro instalaci a „*nfsen.conf*“ je konfigurační soubor pro načtení všech detailů pro instalaci a provoz, jež jsme si předtím upravili. Konfigurační soubor umožňuje správu zdrojů NetFlow, změnu složek pro ukládání statistik, atd. NfSen spustíme pomocí příkazu „*sudo /NfSen složka/bin/nfsen start*“, při změně konfiguračního souboru je třeba provést příkaz „*sudo /NfSen složka/bin/nfsen reconf*“ a ukončení NfSen pomocí příkazu „*sudo /NfSen složka/bin/nfsen stop*“. Webové rozhraní lze spustit po zadaní adresy http://IPserveru/nfsen/nfsen.php.

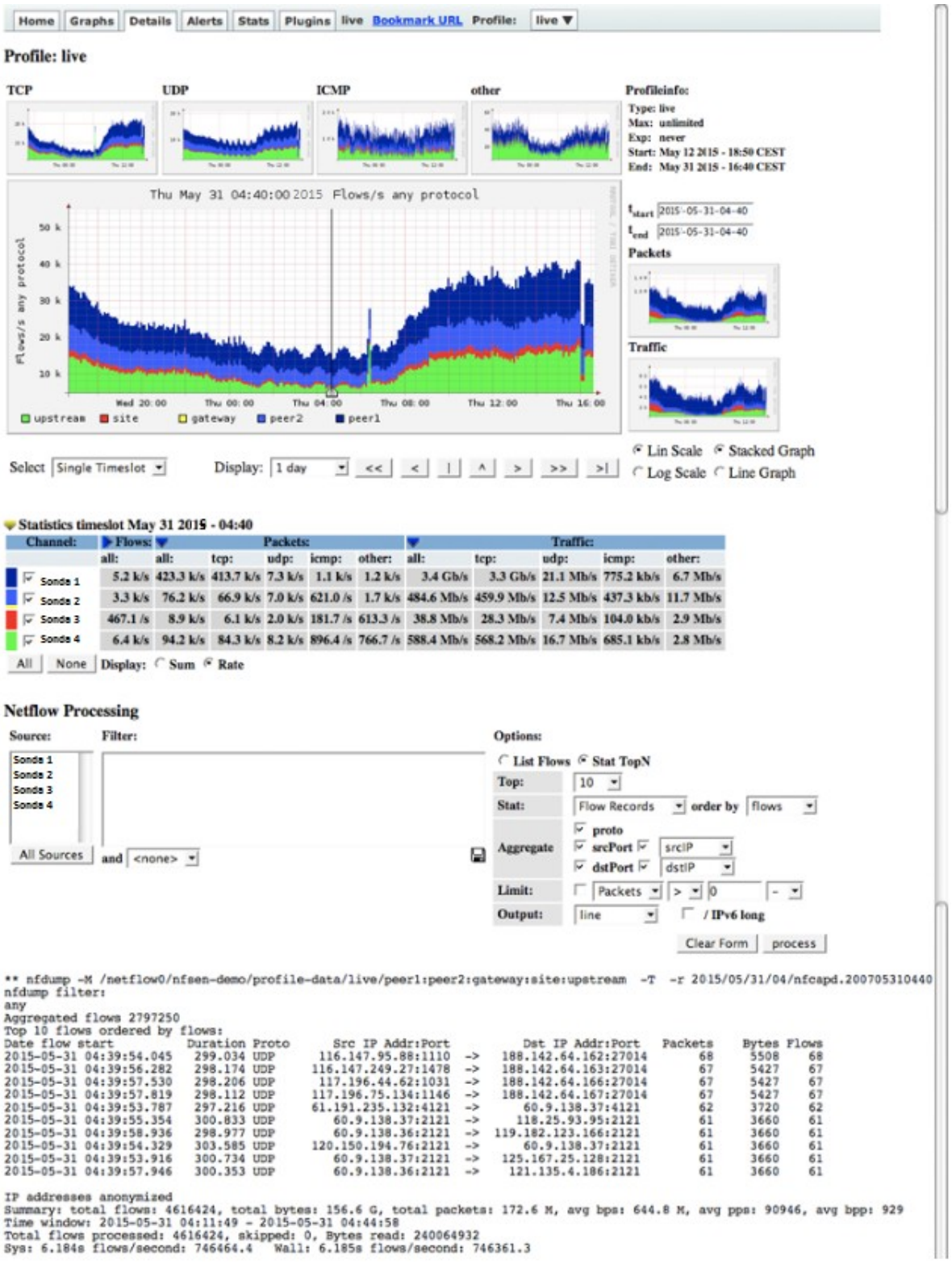
5.1.2 Pohled na NfSen

Při prvním pohledu na úvodní obrazovku NfSen (obrázek 5.1) zprvu uvidíme spoustu nic neříkajících grafů, následně ale zjistíme, že jednotlivé zdroje (sondy) NetFlow dat jsou rozlišeny podle barev, které si můžeme přizpůsobit dle svých preferencí. V jednotlivých řádcích jsou grafy rozlišeny podle délky časové osy, která je ve výchozím stavu nastavena na jeden den, jeden týden a jeden měsíc. Díky tomu můžeme vidět, kdy a jak se určité události opakují, či jestli to nejsou potencionální anomálie. Pokud se podíváme na provoz za celý měsíc, kde se v případě „normálního“ provozu budou události s určitou pravidelností opakovat, ale vyskytne-li se zde určitá odchylka, jistě si zaslouží bližší zkoumání. Zajímavější pohled však nabízejí jednotlivé sloupce, které jsou rozděleny podle toků za sekundu, počtu paketů za sekundu a poslední sloupec zobrazuje počet bitů za sekundu. Na obrázku 5.1 vidíme jistou odchylku v denním přehledu mezi počtem toků za sekundu a počtem paketů za sekundu. Přibližně kolem půlnoci se počet toků snížil, což značí snížení počtu uživatelů, ale naopak počet paketů za sekundu se výrazně zvýšil. To znamená, že malý počet uživatelů generoval velký provoz. Tento provoz může být zapříčiněn například zálohováním na síťový disk v době, kdy nebude omezovat ostatní provoz nebo v horším případě to může znamenat DoS útok na jednu konkretní službu.



Obrázek 5-1: Pohled na hlavní stránku NfSen

Na další záložce „*Details*“ je podrobnější pohled na procházející provoz. Můžeme zde přepínat mezi protokoly TCP/UDP/ICMP a ostatním, které se nám zobrazí na vetším grafu, kde se lze posouvat po časové ose a nastavit její rozsah. Níže na této záložce jsou v přehledné tabulce zobrazeny číselné hodnoty provozu pro jednotlivé protokoly a zdroje dat, které lze také zapnout či vypnout pro samostatné zobrazení v grafu. Další a poslední věcí, která se na této záložce nachází je zobrazení statistik ve stejném formátu jako v příkazové řádce řazených podle různých kriterií a filtrovaných například podle zdrojové, cílové IP adresy a portu, čísla autonomního systému, vstupního či výstupního rozhraní, protokolu atd. Tato záložka je na obrázku 5.2. [12]



Obrázek 5-2: Pohled na záložku "Details" - NfSen

5.2 NfDump

Balíček programů, které pracují s NetFlow daty a jsou ukládány do časově rozdělených souborů. Aplikace běží v příkazové řádce a je porovnatelná s aplikací tcpdump a to i syntaxí . Je napsána v jazyce „C“, což zaručuje její rychlost při zpracování dat. Podporuje NetFlow ve verzích 5, 7 a 9. [13]

Nfdump se skládá z:

- nfcapd – čte NetFlow data ze sítě a ukládá je do souborů každých „n“ minut (typicky každých 5 minut). Pro každý NetFlow zdroj je potřeba jeden nfcapd proces.
- nfdump – čte NetFlow data ze souborů uložených pomocí nfcapd. Syntaxe je podobná jako v tcpdump. Zobrazuje NetFlow data a můžeme vytvářet statistiky seřazené na základě IP adres, portů nebo dalších položek které obsahuje NetFlow tok.
- nfprofile – čte NetFlow data ze souborů uložených pomocí nfcapd. Filtruje NetFlow data podle specifikovaných pravidel a ukládá je pro pozdější použití.
- nfreplay – čte NetFlow data ze souborů uložených pomocí nfcapd a posílá je přes síť pro další zpracování.
- nfclean.pl – skript pro promazání paměti od nepotřebných dat.
- ft2nfdump – převádí data uložena pomocí nástroje flow-tools, aby byla čitelná pro nfdump.

5.3 Suricata IDS

Suricata je na pravidlech založený systém detekce a prevence průniku napsána v jazyce „C“. Vývoj započal v roce 2008 a byli to Matt Jonkman a Viktor Julien, kteří tento program představili světu o rok později. Založili neziskovou organizaci OISF (Open Information Security Foundation), jejíž členové jsou přední světoví experti na bezpečnost a aplikaci neustále vyvíjejí. Software je vyvíjen pod GPLv2 licencí a kód je veřejně dostupný. Je zde zabudovaná podpora pro hardwarovou akceleraci a více vláknové zpracování, díky tomu dosahuje rychlosti zpracování dat až 10G/s v reálné síti. Rozšířené možnosti zpracování zachycených dat je možné pomocí skriptovacího jazyka „Lua“. Můžeme využit pravidla z další podobné aplikace „Snort“, která je nyní provozována společností Cisco. Suricatu IDS je možné provozovat na všech běžných operačních systémech jako jsou Linux, Windows, FreeBSD, OpenBSD, Mac OS a to i sledování v reálném čase s pasivním monitoringem. V nativním módu Suricata IDS nepodporuje zpracování NetFlow. NetFlow podpora je pouze na úrovni výstupů v tomto formátu. Podpora zpracování na více jádrech procesoru a více vlákny, podpora zpracování pomocí GPU, díky tomu dokáže Suricata IDS zpracovat dvakrát větší objem dat na stejně výkonném stroji a za stejný čas jako Snort. [14][15]

Suricata IDS se skládá ze dvou základních částí, a to:

- zařízení na zachycení dat (ve formátu *.pcap), která vstupují a vystupují ze systému,
- pravidla, která určují, jaký typ provozu bude zastaven a vpuštěn dál do sítě.

Možnosti a funkce Suricata IDS:

- dekódování šifrovaného tunelu,
  - Teredo, IP-IP, IPv6-IPv4, IPv4-IPv6, GRE,
- složení původního kompletního toku,
- podpora širokého spektra protokolů,
  - Ipv4, Ipv6, TCP, UDP, SCTP,, ICMPv4, ICMPv6, GRE,
  - Ethernet, PPP, PPPoE, Raw, SSL, VLAN, QinQ, MPL, ERSPAN,
  - HTTP, SSL/TLS, SMB, SMTP, FTP, SSH, DNS, Modbus.
- dekódování HTTP,
  - HTTP log soubory, detekce přenosu souborů,
  - rozdělení detekce jednotlivých HTTP serverů,
  - detekce podle klíčových slov jako jsou:
    - URI, URL, hlavička, cookie, web browser, uživatel, tělo HTTP požadavku a odpovědi
- detekování provozu pomocí:
  - klíčových slov protokolů, pravidel, velikosti či názvu souboru, MD5, a další. [16]

5.3.1 Instalace Suricata IDS

Před samotnou instalací se provádí upgrade systému pomocí příkazů *„sudo apt-get update“* a *„sudo apt-get upgrade“*. Suricata IDS vyžaduje pro svůj běh několik dodatečných balíčků, které nainstalujeme pomocí „*sudo apt-get -y install libpcre3 libpcre3-dbg libpcre3-dev build-essential autoconf automake libtool libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libcap-ng-dev libcap-ng0 make libmagic-dev libjansson-dev libjansson4 pkg-config“*.

Suricata IDS standardně pracuje v módu IDS, pokud ji chceme využívat v IPS módu, nainstalují se další rozšiřující balíčky, *„sudo apt-get -y install libnetfilter-queue-dev libnetfilter-queue1 libnfnetlink-dev libnfnetlink0“*.

Na řadu konečně přichází samotná instalace. Jsou dvě možnosti, jak nainstalovat Suricatu IDS.

První pomocí standardního procesu. Jelikož Suricata IDS není v repositářích běžně rozšířených distribucí, je nutné ji přidat pomocí příkazu *„sudo add-apt-repository ppa:oisf/suricata-stable* a následně nainstalovat samotné Suricata IDS pomocí „*sudo apt-get install suricata“*. V případě této možnosti jsou snadné aktualizace, které se provádí společně s aktualizací samotného systému a ostatních aplikací či balíčků.

Druhá možnost je ruční instalace a stažení jednotlivých součástí z webu distributora ve verzi, kterou požadujeme. S tímto souvisí i případná aktualizace, jež je o poznání náročnější. Pro instalaci musíme stáhnout balíček suricata-3.0.tar.gz, pomocí příkazu *„sudo wget* http://www.openinfosecfoundation.org/download/suricata-3.0.tar.gz*“*, rozbalíme jej pomocí *„sudo tar -xzvf suricata-3.0.tar.gz“* a otevřeme nově vytvořenou složku. Instalace má několik voleb pomocí kterých ovlivníme následné chování a možnosti. V tomto případě jsem použil příkaz *„sudo ./configure*

--enable-nfqueue --prefix=/usr --sysconfdir=/etc –localstatedir=/var“ místo *„sudo ./configure --prefix=/usr --sysconfdir=/etc –localstatedir=/var“* pro možnost využití jako IPS. Dále pokračujeme s příkazy „sudo ./configure --prefix=/usr --sysconfdir=/etc –localstatedir=/var“, „sudo make“, „sudo make install“ a sudo ldconfig, které dokončí instalaci.

Pro podporu výstupů ve formátu NetFlow je nutné toto povolit v konfiguračním souboru suricata.yaml v části „eve-log“ úplně na konec za „- drop, - ssh a - flow“ přidáme „- netflow“. Nicméně toto nám neumožní číst NetFlow záznamy. [17]

5.3.2 Konfigurace a první spuštění Suricata IDS

Po instalaci je třeba provést nastavení zpracování provozu, adresáře pro uložení log souborů a vytvořit pravidla pro zacházení se síťovým provozem. Suricata IDS obsahuje několik základních pravidel, které je možné stáhnout z oficiálních stránek a které fungují velmi dobře. Nicméně pro lepší funkci je třeba tyto pravidla mnohdy upravit, v našem případě si vytvoříme svá vlastní pravidla.

Je nutné ručně vytvořit adresáře pro log soubory a konfigurační soubor. standardních cestách a to *„sudo mkdir /var/log/suricata“* a *„sudo mkdir /etc/suricata.* Tyto cesty se mohou lišit v jednotlivých distribucích linuxu. Není to však důležité, jelikož je lze změnit v konfiguračním souboru. Zkopírujeme několik konfiguračních souborů s úvodním nastavením do adresáře /etc/suricata a to „*sudo cp classification.config /etc/suricata“, „sudo cp reference.config /etc/suricata“* a *„sudo cp suricata.yaml /etc/suricata“*. Posledně jmenovaný obsahuje všechny cesty k jednotlivým pravidlům, konfiguračním souborům, nastavení jaké log soubory se budou generovat, která pravidla se budou využívat, atd.

Suricatu IDS spustíme pomocí příkazu „*sudo suricata -c /etc/suricata/suricata.yaml -i eth0“*. Při spuštění je dobré k spouštěcímu příkazu doplnim volby *„--init-errors-fatal -v“*, diky této volbě uvidíme, která pravidla se načetla a případně důvody, které zapříčinily pád aplikace. Tyto chyby jsou lehce k nalezení na internetu. Nejčastější chyba, se kterou jsem se setkal po každé instalaci bylo špatné zpracování souborů *.pcap síťovou kartou, což lze vyřešit pomocí příkazu *„sudo ethtool -K eth0 gro off“*. Další častá chyba je snaha o načtení neexistujícího pravidla. [18]

5.3.3 GUI Suricata IDS

Suricata IDS umožňuje správu z grafických nástrojů jako je Snorby, Base atd, které jsou provozovány na jiném stroji než samotná Suricata IDS. Nástroje získávají data pro grafy a tabulky z databáze, kterou si sice vytvoří, ale potřebují do ní dostat data, která byla zpracována Suricatou, tuto aktivitu obstarává nástroj Barnyard2. Dalším nástrojem je ELK (Elasticsearch, Logstash a Kibana), jež je určen pro centralizovanou správu, analýzu a zobrazení log souborů. Elasticsearch ukládá a vyhledává log soubory, Logstash tyto log soubory zpracovává a Kibana je webové rozhraní pro zobrazení těchto log souborů.

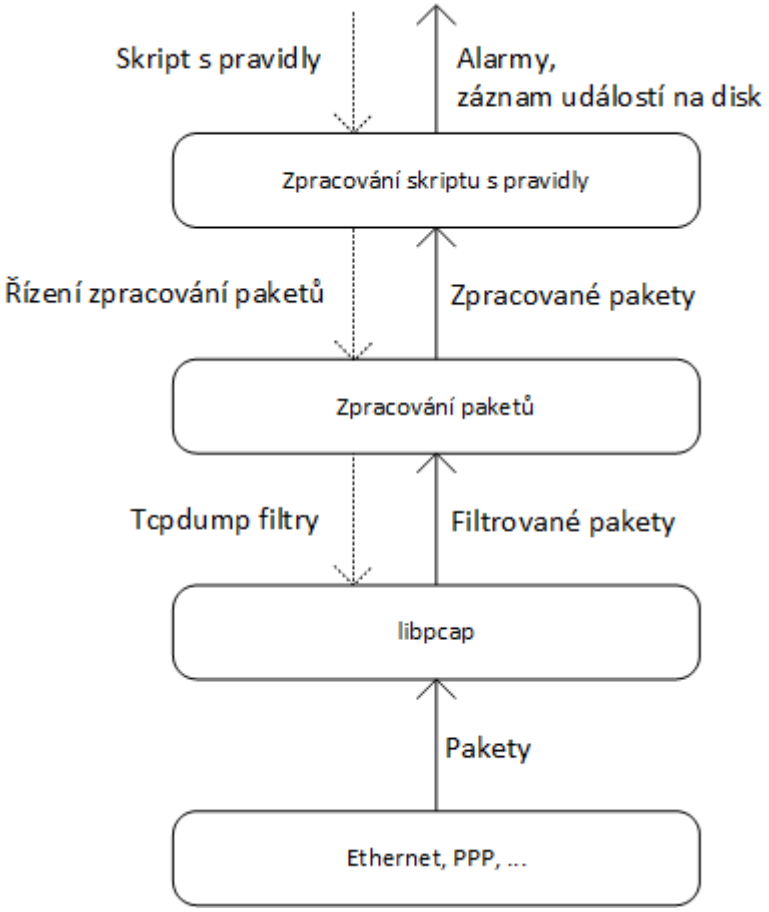
5.4 Bro IDS

Bro IDS je „open-source“ nástroj pro monitorování síťového provozu, založený na Unix systému napsaný v jazyce C++. Vývoj započal v roce 1995 profesor Vern Paxson z Lawrence Berkeley National Laboratory (LBNL) při kalifornské univerzitě Berkeley. Pod jejíž licencí BSD je tento software také veden. Název vychází z románu 1984 (Nineteen Eighty-Four) spisovatele George Orwella, jako připomínka toh,o že monitorování jde ruku v ruce s potencionálním narušením soukromí. Dnes sponzoruje vývoj Bro IDS National Science Foundation (NSF) a zastřešen International Computer Science Institute (ICSI), působící v rámci univerzity Berkeley, kde se Vern Paxton stále podílí na vývoji Bro IDS jako hlavní vývojář. Bro IDS umožňuje spojit více počítačů s běžícím Bro IDS do clusteru jako jeden celek pro dosažení vyšší rychlosti zpracovaných dat s propustností až 100Gb/s. Pro možnosti centrální správy tohoto clusteru, stejně tak jedné instance Bro IDS, slouží nástroj „BroControl“, který poskytuje synchronizaci jednotlivých instancí, nastavení a přístup k záznamům o aktivitách. Bro IDS provozuje mnoho institucí pro ochranu jejich počítačové infrastruktury a patří mezi ně: university, výzkumné laboratoře, superpočítačová centra, velké korporace a další. Bro IDS jako mnoho síťových nástrojů založených na Unix systému používá “libpcap”, jako součást své architektury pro zpracování dat na síti. Bro IDS může fungovat jako pasivní monitorovací nástroj bez přidělené IP adresy, připojen k portu přepínače s nastaveným span módem. Bro IDS má vestavěnou podporu pro analyzování a detekování, do níž patří extrahování souborů z HTTP spojení, detekování malware pomocí externích registrů, reportování zranitelností jednotlivých verzí software komunikujícího na síti, identifikace webových aplikací a případně jejich blokace, detekování SSH brute-force, ověřování SSL certifikátů a mnohem více. Bro IDS generuje log soubory s dobře čitelnými a standardizovanými údaji pro další případné zpravování softwarem třetích stran. Bro IDS je velmi mocný nástroj, který je levnou a rovnocennou alternativou k placeným a velmi drahým řešením. [19]

5.4.1 Architektura Bro IDS

Bro IDS může být využíváno na různých typech sítí díky podpoře libpcap. Typická architektura systému Bro IDS, jež je konceptuálně rozdělen na několika vrstev samostatných bloků, viz. Obrázek 5.3.

Toto rozdělení má svůj účel a přispívá k lepšímu a méně náročnému zpracování dat. Ze sítě si všechna data převezme knihovna „libpcap“ a dále je předá bloku pro zpracování dat, potom bloku pro zpracování podle skriptu s pravidly, na základě kterého jsou vyhlašovány poplachy a tyto ukládány na disk. My jako správce díky těmto alarmům, můžeme lépe optimalizovat skript s pravidly, který dále upraví zpracovávané data a aplikují se filtry pro knihovnu „libpcap“, to znamená, že vyšší vrstvy mají zpětně, díky těmto filtrům, na starosti menší objem zpracovávaných dat.



Obrázek 5-3: Architektura systému Bro IDS

- **Libpcap**

Nad samotnou sítí se nachází knihovna pro odchytávání síťového provozu, která je využívaná a vyvíjená společně s „tcpdump“. Tato knihovna izoluje Bro od technologii samotné sítě (jako je: Ethernet, FDDI, PPP, atd.), což usnadňuje portování Bro na jiné Unix systémy, či novější hardware. Také to znamená že Bro umožňuje off-line analýzu dat, jež jsou uložená pomocí tohoto formátu. Další výhodou „libpcap“ je, že pokud má operační systém implementovánu dobrou podporu pro filtrování paketů, jako je např. BPF (Berkeley Packet Filter), pak je schopen efektivně filtrovat pakety již na úrovní jádra. Tudíž, raději než posílat všechna data do vyšších vrstev až na úroveň uživatele, může být většina dat vyřazena hned po přijetí na síťové kartě v případě, že filtr je takto nastaven. Díky tomuto filtrování dochází k rychlejšímu zpracování zbylých dat.

- **Zpracování paketů**

Po vyfiltrování jsou pakety předány další vrstvě a tou je „Zpracování paketů“. Tato vrstva kontroluje jestli jsou pakety v pořádku pomocí pole CRC v hlavičce paketu. Pokud tato kontrola neprojde bez chyby je tento paket automaticky zahozen a Bro vygeneruje událost indikující zahození paketu. Bro také v této fázi skládá jednotlivé IP datagramy do jednoho celku pro výslednou analýzu.

Pokud je kontrola uspěšná, blok zkontroluje, jestli n-tice IP adres a TCP/UDP portů již existuje, pokud ne tak vytvoří nový záznam, pokud ano přiřadí jej k již existujícímu. Bro ukládá celý paket i s daty pokud jej bude dále zpracovávat, hlavičku pouze pokud se jedná o pakety

SYN/FIN/RST a nebo jej přeskočí, pokud není potřeba paket ukládat. Tento blok porovnává pakety s pravidly pro zpracování a na základě těchto pravidel vyhodnocuje, jestli vygeneruje nějakou událost.

- **Zpracování skriptu s pravidly**

Jakmile zpracování paketu skončí, tento blok zkontroluje, zda byly vytvořeny nějaké události, které jsou dále zpravovány pomocí FIFO fronty. Klíčovou vlastností Bro je rozdíl mezi generováním událostí a reakcí na tyto události, proto jsou v diagramu uvedeny v odlišných blocích. Skripty jsou napsány v jazyce „Bro“. Každá událost, která se dostala až do tohoto bloku je zpracována na základě těchto skriptů. Bro skript může generovat nové události, vytvářet log soubory (podobné Unix *syslog*) nebo ukládat data na disk. Pomocí těchto událostí, které jsou interpretovány uživateli, upravujeme tento skript pro jejich lepší a přesnější generování.

5.4.2 Skriptovací jazyk *Bro*

Skripty s pravidly jsou napsány ve speciálním skriptovacím jazyce *Bro*. Jazyk je vytvořen tak, aby přímo pracoval s hostname, IP adresami, čísly portů, atd. Bro podporuje několik datových typů z běžných programovacích jazyků, jako jsou *bool, int, count, string* a mnohé další. Jsou zde také netradiční datové typy. Hodnota *time* určuje absolutní čas, typ *port* koresponduje s TCP nebo UDP číslem portu a zapisuje se jako číslo portu „/tcp“ nebo „/udp“, například pro http bude hodnota 80/tcp. Jako další je typ *addr* pro IP adresu, která se zadává v klasickém formátu (192.168.1.1). Pro označení zdrojové IP adresy se využívá elementů *orig_h:* stejně tak pro cílovou IP adresu se využívá *resp_h:*. Pro porty se využívá elementů *orig_p:* a *resp_p:*. Pokud chceme definovat komunikaci se zdrojovou a cílovou IP adresou a portem na webový server bude to vypadat následovně:

```
type conn_id record {
    orig_h: 192.168.1.1;
    orig_p: 9384;
    resp_h: 192.168.1.2;
    resp_p: 80;
```

5.4.3 Funkce Bro IDS

- Nasazení:
  - lze provozovat na všech Unix-like systémech (Linux, FreeBSD, MacOS),
  - pasivní analýza dat ze souboru nebo real-time monitorovaní,
  - podpora libpcap pro odchytávání paketů,
  - podpora seskupení do clusteru pro vyšší výkon,
  - open-source pod BSD licencí.
- Analýza:
  - komplexní záznamy pro offline analýzu a forenzní analýzu,
  - analýza protokolů aplikační vrstvy (DNS, FTP, HTTP, IRC, SMTP, SSH, SSL, …),

- analýza obsahu souboru přenášeného z aplikační vrstvy, obsahující MD5/SHA1 pro kontrolu,
- komplexní podpora IPv6,
- detekování tunelu a jeho analýza (Teredo, Ayiya, GTPv1), Bro dešifruje tunel pomocí soukromého klíče (v případě že jej obsahuje) a data analyzuje jako by tam žádný tunel nebyl,
- podpora signatur z klasických IDS, není to však primární zdroj pravidel,
- standardizované log soubory

- Skriptovací jazyk:
  - Turing-complete skriptovací jazyk k vyjádření libovolných úkolu pro analýzu,
  - model založený na událostech,
  - specifické datové typy jako je IP adresa, číslo portu, atd.,
  - externí knihovna jazyka C pro komunikaci s externími programovacími jazyky, jako je Perl, Python a Ruby,
  - schopnost spouštět externí procesy pomocí *Bro* skriptovacího jazyka. [19][20]

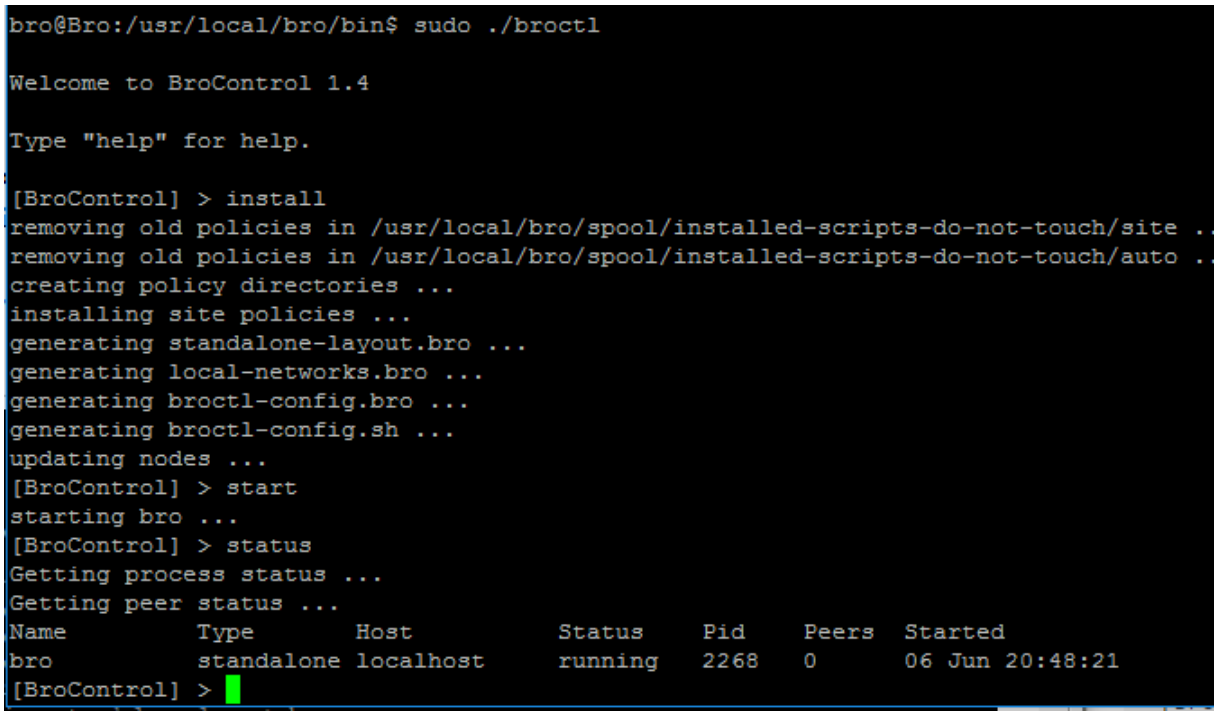
5.4.4 Instalace Bro IDS

Před samotnou instalací je třeba provést upgrade systému pomocí příkazů *„sudo apt-get update“* a *„sudo apt-get upgrade“*. Bro IDS vyžaduje pro svůj běh několik dodatečných balíčků, které nainstalujeme pomocí „*sudo apt-get -y install cmake make gcc g++ flex bison libpcap-dev libssl-dev python-dev swig zlib1g-dev“* volitelně, potom balíček pro geo lokaci či odesílání e-mailu při nastavené události. Pokud máme nainstalovány všechny dodatečné balíčky můžeme stáhnout samotné Bro IDS pomocí příkazu *„sudo wget https://www.bro.org/downloads/release/bro-2.4.1.tar.gz“* a následně rozbalit pomocí „*sudo tar -xvzf bro-2.4.1.tar.gz“*. V rozbalené složce „bro-2.4.1“ spustíme příkaz „*sudo ./configure --prefix=/nsm/bro“* pro konfiguraci následné instalace. Dále spustíme příkaz „*sudo make“* pro sestavení a kompilaci Bro IDS, která trvá poměrně dlouho dobu a na virtualizovaném stroji s jedním procesorem a 1GB RAM trvala přibližně přes půl hodiny. Pro dokončení už pouze instalujeme pomocí příkazu „*sudo make install“*. [21]

5.4.5 Konfigurace a první spuštění Bro IDS

Než spustíme poprvé Bro IDS musíme upravit několik konfiguračních souborů umístěných ve složce etc/, která se nachází ve stejném umístění instalace samotného programu. Tyto soubory jsou především „node.cfg“, kde je nutno upravit síťové porty, na nichž chceme odchytávat žádaný provoz, „networks.cfg“, kde definujeme lokální subnet a poslední soubor je „broctl.cfg“, kde nastavíme emailovou adresu pro odesílání hlášení o událostech.

První spuštění provedeme pomocí příkazu „*broctl“*, který musíme v případě, že jsme nenastavili „PATH“, spustit ze složky instalovaného Bro IDS a dostaneme vlastní příkazovou řádku pro Bro IDS, jež vidíme na obrázku 5.4. Rovněž je dobré při každém spuštění zadat příkaz „*install“*, který provede znovu načtení konfigurací a politik. V této fázi program ještě neodchytává žádný provoz, zadáme tedy příkaz „*start“* a analogicky odchytávání zastavíme pomocí „*stop“*.



Obrázek 5-4: Příkazová řádka BroControl

5.4.6 Log soubory

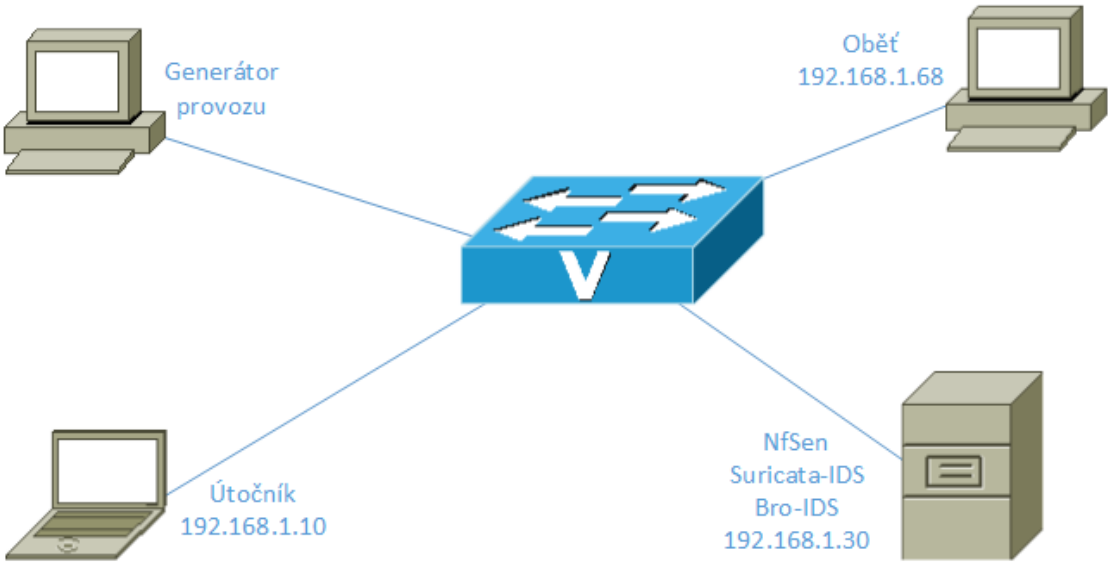
Log soubory můžeme rozdělit na dvě skupiny aktuálně odchytávané a již odchycené. Již odchycené jsou řazeny do adresářů dle data odchycení a následně jednotlivé soubory dle času. Příklad log souborů viz. Obrázek 5.5.

Obrázek 5-5: Příklad log souborů

Například záznam „*http.19:41:46-20:00:00.log.gz*“ obsahuje přesný čas, zdrojovou/cílovou IP adresu/port, typ dotazu (GET, OPTIONS, …), typ prohlížeče a další pro každý http paket. Díky těmto údajům jsme schopni zjistit, co přesně se na síti dělo a můžeme z nich poskládat původní komunikaci.

6 Testování anomálií

Pro testování jsem zvolil jednoduchou topologii, kde se útočník a oběť nachází v jednom subnetu. Všechny PC jsou instalovány s OS „Ubuntu server 12.04“ nebo „Ubuntu server 14.04“. Konfigurace jednotlivých PC jsou popsány níže. Stroje běží v aplikaci „VirtualBox“.



Obrázek 6-1: Topologie pro testování

Pro generování „normálního“ provozu využijeme nástroj „Ostinato“, který běží na nezávislé stanici. Pro simulaci jsem vybral DoS útok a skenování portů.

- **DoS**

Útok vedený na server, který zapříčiní jeho nedostupnost, je vedený z jednoho počítače na druhý, je však nutné, aby útočník měl rychlejší připojení k internetu než oběť jinak by útok neměl smysl. Verze, kdy je útok veden z mnoha (až tisíců) počítačů pod kontrolou útočníka pro zvýšení síly na jeden počítač je tzv. „Distribuovaný DoS“. Existuje celá řada nástrojů pro provedení DoS útoku, pomocí nichž je tento útok schopen provést i naprostý laik, stejně tak existuje celá řada způsobů provedení tohoto útoku. Zmíním zde dva nejčastější, a to:

- **ICMP flood** – Útok je založen na tom, že útočník odešle obrovské množství ICMP paketů na broadcast adresu s IP adresou oběti jako zdrojovou, díky tomu všichni účastníci komunikace odpovídají na IP adresu oběti a tím ji zahltí.
- **SYN flood** - Útočník nebo počítače pod jeho kontrolou odešlou na IP adresu oběti TCP/SYN paket. Server v domění, že se jedná o žádost o spojení se jej bude snažit navázat a odešle TCP/SYN-ACK paket, čeká na potvrzení od útočníka, ale to nepřichází. Server drží polo otevřené spojení, po určitou dobu mezi tím přijde další a další požadavek

o přístup a server vyčerpá počet dostupných spojení, které zvládne obsloužit a tím se stane nedostupným pro legitimní požadavky. Útok může být veden na konkrétní porty.

Obranou proti těmto útokům by měl být v našem případě dobře nastavený IDS/IPS systém. Existuje ale i mnoho dalších způsobů, jak se bránit od odklonění útoku do tzv. „černé díry“ přes směrovače, přepínače a firewally.

Tento útok dobře poslouží jako anomálie vybočující z onoho normálního provozu, při němž očekáváme určitý typ provozu, který se může měnit v průběhu času a v určitých intervalech se opakovat. Při onom útoku zaznamenáme prudký nárůst provozu a budeme čekat nějakou reakci od testovaných nástrojů v podobě oznámení útoku.

- **Skenování portů**

Je metoda pro ověření dostupnosti jednotlivých portu jak TCP tak UDP. Metoda je založena na posílání paketů s měnícím se cílovým portem, tzn. pokud chceme prozkoumat tisíc portů pošleme tisíc paketů. Skenování portů bylo původně pro administrátory pro ověření určité služby, ale stalo se velmi populární metodou i pro méně dobromyslné aktivity. Skenování portů může předcházet dalšímu útoku, který se zaměří na dostupné služby. Pokud provozujeme server je lepší všechny nepoužívané porty uzavřít, aby útočníci měli menší šanci na napadení, zbývající námi používané porty můžeme lépe zabezpečit, jelikož o nich budeme mít lepší přehled. Existuje mnoho typů skenování portů, uvedu však jen ty více používané:

- **TCP sken** – skenovací nástroj naváže TCP komunikaci a dokončí celý „3-way handshake“ a následně spojení uzavře aby se vyhnul detekci jako DoS útok.
- **SYN sken** – skenovací nástroj naváže komunikaci pomocí TCP/SYN paketu, počká na přijetí TCP/SYN-ACK paketu a odešle TCP/RST paket pro ukončení komunikace. Tento způsob je méně náročný na vzdálený server, jelikož spojení uzavřeme.
- **UDP sken** – na rozdíl od TCP skenu je spojení nespojové, můžeme ale využít toho, že vzdálený server v případě, že je port zavřený, odpoví zprávou „ICMP unreachable“ , port není dostupný.

Tyto útoky budou provedeny vůči stanici „oběť“, která by v reálné situaci mohla zastávat funkci sondy, pokud by byla tranzitní. Na stanici bude instalován nástroj „fprobe“ a „tcpdump“ pro odchycení provozu a následně odeslán k analýze testovanými nástroji.

6.1 Útočník

PC útočníka je provozováno s operačním systémem „Kali“,jež po instalaci aktualizujeme pomocí příkazů „*sudo aptitude update & sudo aptitude upgrade“*. Tento operační systém je založen na distribuci „Debian“. Jde tedy o open-source a obsahuje řadu nástrojů pro forenzní analýzu a penetrační testy, hodí se jak pro nezákonné účely, tedy napadení systému či sítě oběti, tak pro techniky, kteří provádí audit na předem domluveném systému či síti.

Pro uskutečnění útoku využijeme pouze dva z více než tří stovek nástrojů, jež „Kali“ nabízí. První z nich je „hping3“, který nabízí bohaté možnosti přepínačů a je určen především pro DoS útok na specifikovaný port oběti. Druhým nástrojem bude „Nmap“ případně jeho grafická verze „Zenmap“,

který je pro změnu určen, pro ověření dostupnosti portů, tedy jejich skenování. Nástroj „Zenmap“ není ve standardní výbavě „Kali“ , tudíž jej nutně musíme doinstalovat pomocí příkazu „*sudo aptitude install zenmap“.*

6.1.1 Útok

DoS útok bude probíhat ve třech, pěti minutových intervalech ve vzorku cca čtyř hodin běžného provozu. Jak jsem uvedl dříve, DoS útok bude proveden za pomocí nástroje „hping3“ a použití příkazu „*hping3 -S -p 80 –flood –rand-source -d 1400192.168.1.68“,* kde *„-S“* je odesílání pouze TCP/SYN paketů, „-p 80“, s čímž budeme všechny pakety odesílat na port webového serveru, „--fast“ bude odesílat deset paketů za sekundu, „--faster“ bude odesílat mnohem více paketů než „fast“ *„--flood“* bude odesílat co nejvíc paketů, kolik dovolí propustnost útočníkovy sítě, „--rand-source“ uvede náhodnou zdrojovou IP adresu v odeslaném paketu, tudíž oběť nepozná odkud je útok veden, „-d 1400”, který určuje velikost odesílaného paketu a nakonec uvedeme IP adresu oběti. Použijeme tři příkazy, každý pro jeden útok, jehož intenzita se bude stupňovat.

První útok bude nejméně intenzivní a bude odesílat 10 paketů za sekundu, což v běžném provozu není vůbec poznat a za oněch 5 minut útoku nám to udělá cca přes 3 tisíc paketů. Průběh útoku vidíme na obrázku 6.2

```
root@kali:~# hping3 -S --fast --rand-source -p 80 -d 1400 192.168.1.68
HPING 192.168.1.68 (eth0 192.168.1.68): S set, 40 headers + 1400 data bytes
^C
--- 192.168.1.68 hping statistic ---
3435 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@kali:~#
```

Obrázek 6-2: DoS útok 1

Druhý útok je již mnohem intenzivnější, jak můžeme vidět na počtu odeslaných paketů za zhruba stejnou dobu, jsou to necelé 2 miliony, což je cca 6500 paketů za sekundu, což je znatelný nárůst. Obrázek 6.3

```
root@kali:~# hping3 -S --faster --rand-source -p 80 -d 1400 192.168.1.68
HPING 192.168.1.68 (eth0 192.168.1.68): S set, 40 headers + 1400 data bytes
^C
--- 192.168.1.68 hping statistic ---
1951966 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@kali:~#
```

Obrázek 6-3: DoS útok 2

Při posledním a nejintenzivnějším útoku jsem přenesl cca 2 700 000 paketů, o necelou polovinu více než v předchozím případě a není to tak obrovský nárůst jako mezi prvními dvěma útoky. V případě rychlejší linky mezi útočníkem a obětí a reálnými počítači, by počet paketů mohl být vyšší, jelikož virtualizace na běžném počítači není stavěná na takový obrovský přenos dat, může to být tedy její limit viz. Obrázek 6.4. Nakonec provedeme ještě jeden velmi krátký minutový útok stejné intenzity.

```
root@kali:~# hping3 -S --flood --rand-source -p 80 -d 1400 192.168.1.68
HPING 192.168.1.68 (eth0 192.168.1.68): S set, 40 headers + 1400 data bytes
hping in flood mode, no replies will be shown
^C
^C
--- 192.168.1.68 hping statistic ---
2698363 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@kali:~#
```

Obrázek 6-4: DoS útok 2

Druhý útok provedeme pomocí nástroje „Nmap“ a budeme skenovat všechny otevřené porty na cílovém serveru. Útok provedeme dvakrát v inkriminované hodiny, jednou pouze základní sken všech portů a v druhém případě to bude intenzivní skenování do hloubky.

První skenování proběhlo velmi rychle v řádu jednotek sekund (v tomto případě dokonce jen 1.56 sekundy), nástroj pouze ověří všechny TCP porty, jestli jsou otevřené, jak můžeme vidět na obrázku 6.5. Výpis je krátký, protože na počítači oběti jsou otevřeny pouze dva porty a to pro SSH a HTTP komunikaci. Zjistili jsme také jakou MAC adresu má oběť ,což můžeme použít v budoucnu pro další útoky.

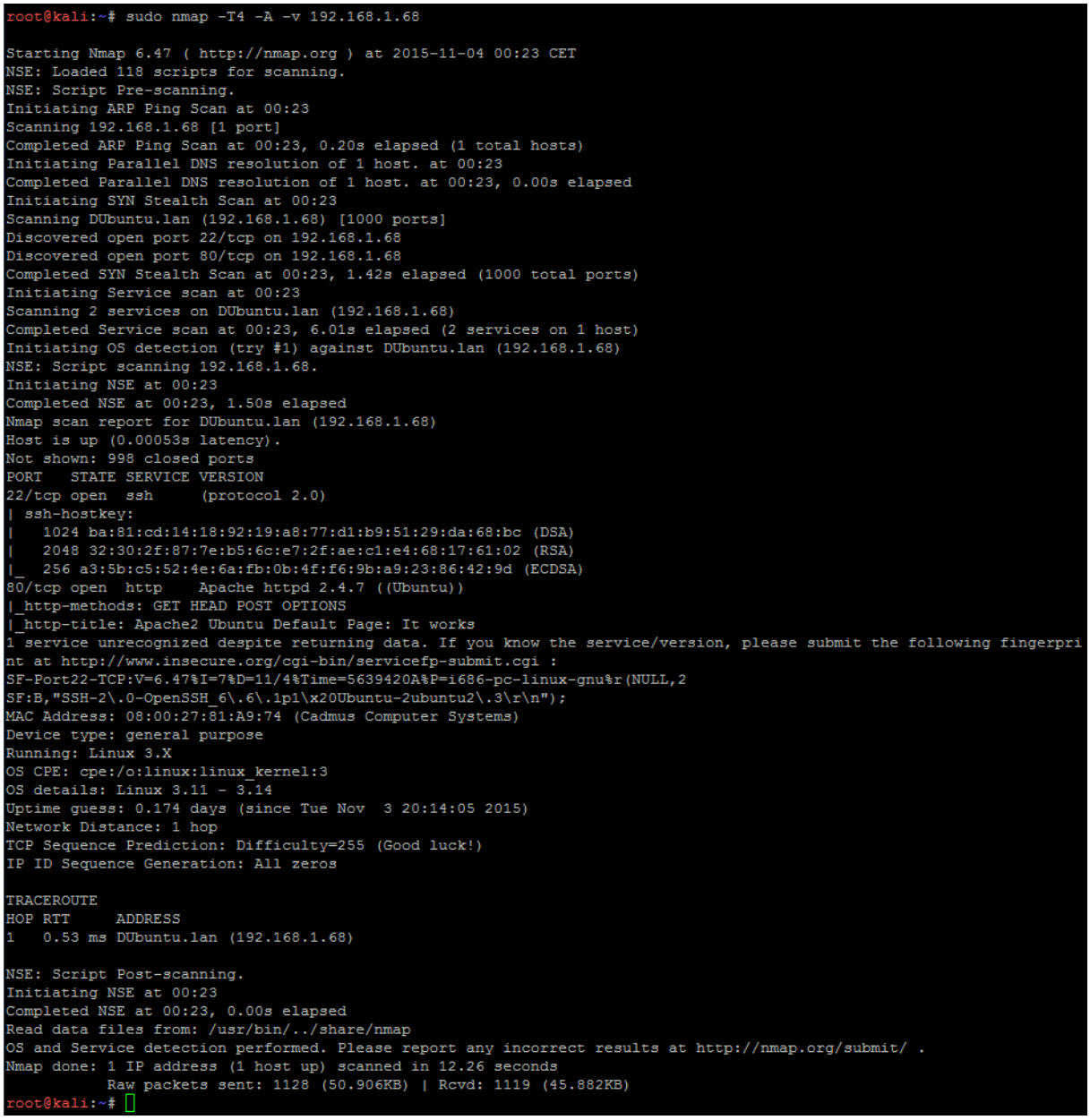
```
root@kali:~# sudo nmap -T4 -F 192.168.1.68

Starting Nmap 6.47 ( http://nmap.org ) at 2015-11-04 00:26 CET
Nmap scan report for DUbuntu.lan (192.168.1.68)
Host is up (0.0035s latency).
Not shown: 98 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:81:A9:74 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 1.54 seconds
root@kali:~#
```

Obrázek 6-5: Rychlé skenování oběti

Druhé skenování je delší a trvá 12.26 sekund, což není i tak moc. Najdeme zde však mnohem podrobnější informace. Z výpisu viz. Obrázek 6.6, je patrné, že typ skenování je „SYN Stealth Scan“, který jsme si popsali již dříve. V tomto případě podrobnějšího skenu, proběhne jeden základní pro ověření dostupnosti stanic a následně druhý pro ověření otevřených portů. Druhý intenzivnější sken proběhne pouze na dostupných stanicích. Díky tomuto zjistíme typ a verzi operačního systému, jak dlouho je stanice zapnutá a samozřejmě detaily o otevřených portech. O SSH portu zjistíme verzi, která je použita (v našem případě je to verze 2.0), veřejné SSH klíče, jak pro šifrování 1024 bit, tak 2048 bit a typ SSH serveru. Stejně tak vidíme typ HTTP serveru a jeho verzi (Apache httpd 2.4.7), titulek úvodní stránky v prohlížeči a příkazy, jež jsem vůči tomuto serveru ověřil, že fungují (GET, HEAD, POST, OPTIONS) a jsou běžně používané při prohlížení webových stránek. [23]



Obrázek 6-6: Intenzivní skenování oběti

6.2 Oběť

PC oběti je nainstalováno s operačním systémem „Ubuntu server 14.04“, který po instalaci aktualizujeme pomocí příkazů „*sudo aptitude update & sudo aptitude upgrade“*. Následně nainstalujeme aplikaci pro odchytávání provozu, což budou „tcpdump“ a „fprobe“, to provedeme pomocí příkazů „*sudo aptitude install tcpdump“* a „*sudo aptitude install fprobe“* :

- tcpdump

Tento nástroj slouží k odchycení provozu na síti a jeho zobrazení na obrazovce. Pomocí něj lze odchycený provoz také uložit ve formátu pcap a samozřejmě z něj i číst. Jeho verzí s grafickým rozhraním je „Wireshark“. „Tcpdump“ nabízí obrovské možnosti pro filtrování zobrazovaných dat

pomocí přepínačů, primárně je určen pro Unix operační systémy a využívá knihovnu „libpcap.“ Data odchycená tímto nástrojem budeme odesílat na server k dalšímu zpracování. V případě sondy by odchycení a odesílání bylo mnohem jednodušší. Jelikož my použijeme jako sondu PC se systémem Linux, odchytíme hodinu veškerého provozu a následně jej odešleme.

- pro odchycení dat: „*sudo tcpdump -i eth0 -w /data/utok.pcap*“

- pro odeslání na server: „*scp /data/utok.pcap uživatel@IPserveru:/data/*“

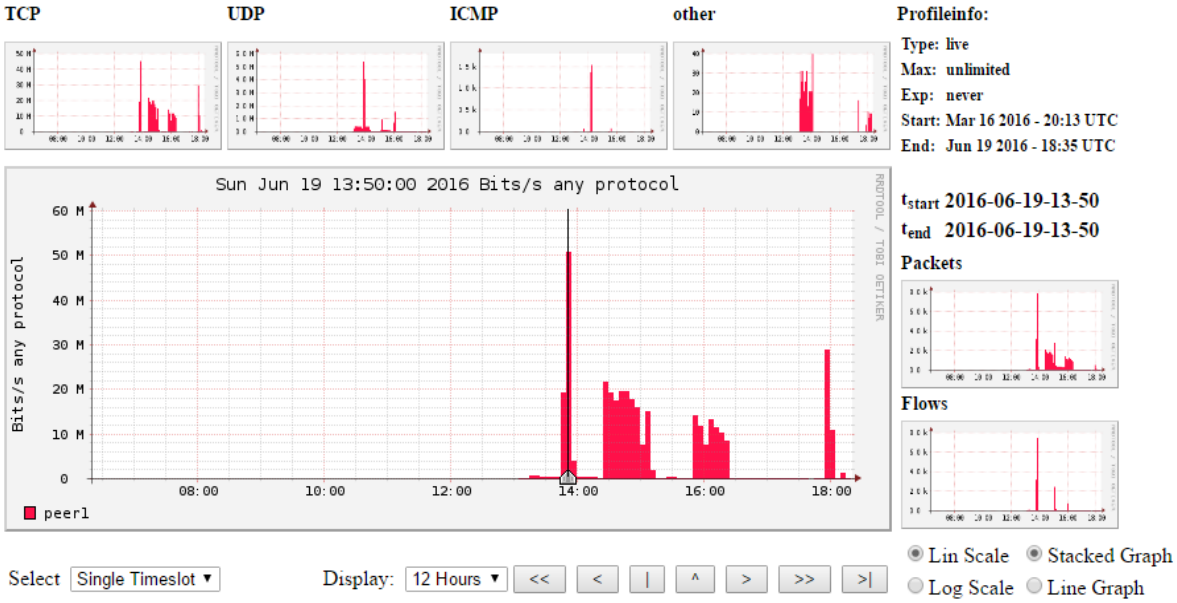
- fprobe

Tento nástroj slouží k odchycení provozu na síti a jeho následného uložení ve formátu NetFlow, v kterém je také odesílán na server k dalšímu zpracování. Pomocí přepínačů, kterých má tento program vcelku dost, můžeme nastavit verzi exportovaného NetFlow, rozdělit odesílaní NetFlow mezi více serverů, specifikovat protokol a službu, kterou chce sledovat a mnohem více. V našem případě nastavíme odesílání na server 192.168.1.30 s portem 9996. Vzhledem k tomu, že chceme sledovat všechen procházející provoz nebudeme jej nijak omezovat či specifikovat, toto provedeme pomocí následujícího příkazu. [22]

„*fprobe -i eth0* 192.168.1.30:9996“

6.3 NfSen

Nástroj NfSen jsem vybral z toho důvodu, protože jako jediný nativně podporuje netflow. NfSen zobrazuje veškerý provoz přehledně v grafech rozdělených podle časových rozpětí a podle typu provozu, kde si ještě můžeme vybrat mezi pakety/s, bity/s či toky/s.

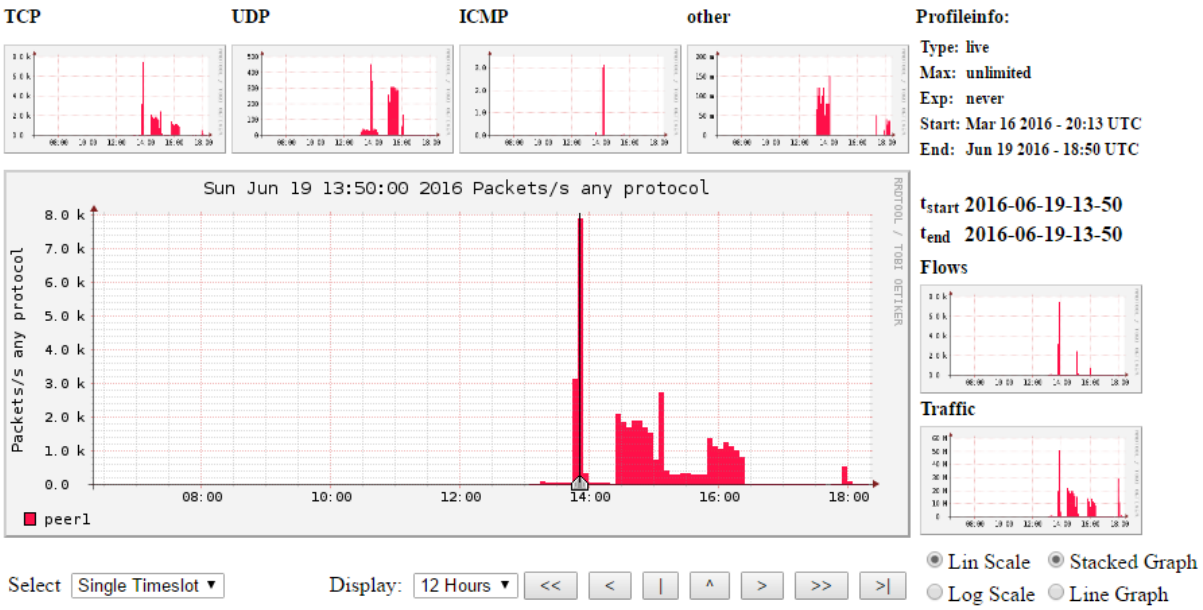


Obrázek 6-7: NfSen – Bit/s

```
** nfdump -M /data/nfsen/profiles-data/live/peer1  -T  -r 2016/06/19/nfcapd.201606191350 -n 100 -s ip/bytes
nfdump filter:
any
Top 100 IP Addr ordered by bytes:
Date first seen          Duration Proto           IP Addr    Flows(%)     Packets(%)       Bytes(%)         pps      bps   bpp
2016-06-19 13:46:55.000   408.189 any       192.168.1.68    2.2 M(99.9)    2.4 M(99.9)    1.9 G(99.8)     5796   37.2 M   801
2016-06-19 13:46:55.000   358.722 any     104.155.10.158        9( 0.0)    132214( 5.6)   197.2 M(10.4)     368    4.4 M  1491
2016-06-19 13:49:36.515   250.242 any     192.168.1.179       52( 0.0)       52( 0.0)     69285( 0.0)       0     2214  1332
2016-06-19 13:48:45.584   299.927 any       192.168.1.5       57( 0.0)      111( 0.0)     61617( 0.0)       0     1643   555
2016-06-19 13:48:54.863   296.097 any     192.168.1.110       46( 0.0)       46( 0.0)     61063( 0.0)       0     1649  1327
2016-06-19 13:48:53.281   288.192 any     192.168.1.188       43( 0.0)       43( 0.0)     56774( 0.0)       0     1576  1320
2016-06-19 13:49:01.039   274.062 any     192.168.1.251       41( 0.0)       41( 0.0)     54879( 0.0)       0     1601  1338
2016-06-19 13:48:52.043   286.898 any     192.168.1.191       40( 0.0)       40( 0.0)     54021( 0.0)       0     1506  1350
2016-06-19 13:48:47.137   285.236 any      192.168.1.70       40( 0.0)       40( 0.0)     53036( 0.0)       0     1487  1325
2016-06-19 13:48:48.622   301.217 any     192.168.1.242       39( 0.0)       39( 0.0)     52133( 0.0)       0     1384  1336
2016-06-19 13:49:08.341   266.860 any     192.168.1.206       36( 0.0)       38( 0.0)     50655( 0.0)       0     1518  1333
2016-06-19 13:48:58.677   275.112 any      192.168.1.86       35( 0.0)       37( 0.0)     48771( 0.0)       0     1418  1318
2016-06-19 13:49:11.978   276.729 any     192.168.1.144       35( 0.0)       35( 0.0)     46910( 0.0)       0     1356  1340
2016-06-19 13:48:50.382   288.962 any     192.168.1.243       34( 0.0)       35( 0.0)     46707( 0.0)       0     1293  1334
2016-06-19 13:49:57.837   222.617 any     192.168.1.167       34( 0.0)       34( 0.0)     45545( 0.0)       0     1636  1339
2016-06-19 13:48:50.852   198.594 any     192.168.1.151       33( 0.0)       33( 0.0)     44459( 0.0)       0     1790  1347
2016-06-19 13:49:22.881   257.364 any     192.168.1.205       33( 0.0)       33( 0.0)     44388( 0.0)       0     1379  1345
2016-06-19 13:49:33.451   256.388 any     192.168.1.246       33( 0.0)       33( 0.0)     44247( 0.0)       0     1380  1340
2016-06-19 13:49:09.343   266.159 any     192.168.1.254       31( 0.0)       33( 0.0)     44104( 0.0)       0     1325  1336
2016-06-19 13:48:49.036   290.209 any     192.168.1.171       33( 0.0)       33( 0.0)     44002( 0.0)       0     1212  1333
```

Obrázek 6-8: nfdump - textový výpis bit/s

Při pohledu na provoz na obrázku 6.7 vidíme velké odchylky, které mohou znamenat útok nebo chybu, měli bychom je tedy blíže prozkoumat. Na velkém grafu uprostřed jsou zobrazeny bit/s pro všechny protokoly jak TCP, UDP tak ICMP a další abychom viděli veškerý typ provozu. Při pochybnostech co může stát za oním zvýšeným provozem nám pomůže textový výpis na obrázku 6.8. Vidíme že největší provoz je na počítači na, který jsme útočili, což je celkem logické, jelikož se na každý TCP/SYN paket snaží odpovědět. Vzhledem k tomu, že jsem při útoku zadal náhodné zdrojové adresy nelze určit, ze kterého počítače byl útok veden a vidíme, že se provoz rozdělil mezi mnoho stanic, přičemž dohromady tvoří velký objem dat. Bohužel narůstající intenzita útoku se neprojevuje jako narůstající výchylka v grafu, to může výt způsobeno zahazováním paketu jádrem.



Obrázek 6-9: NfSen - pakety/s

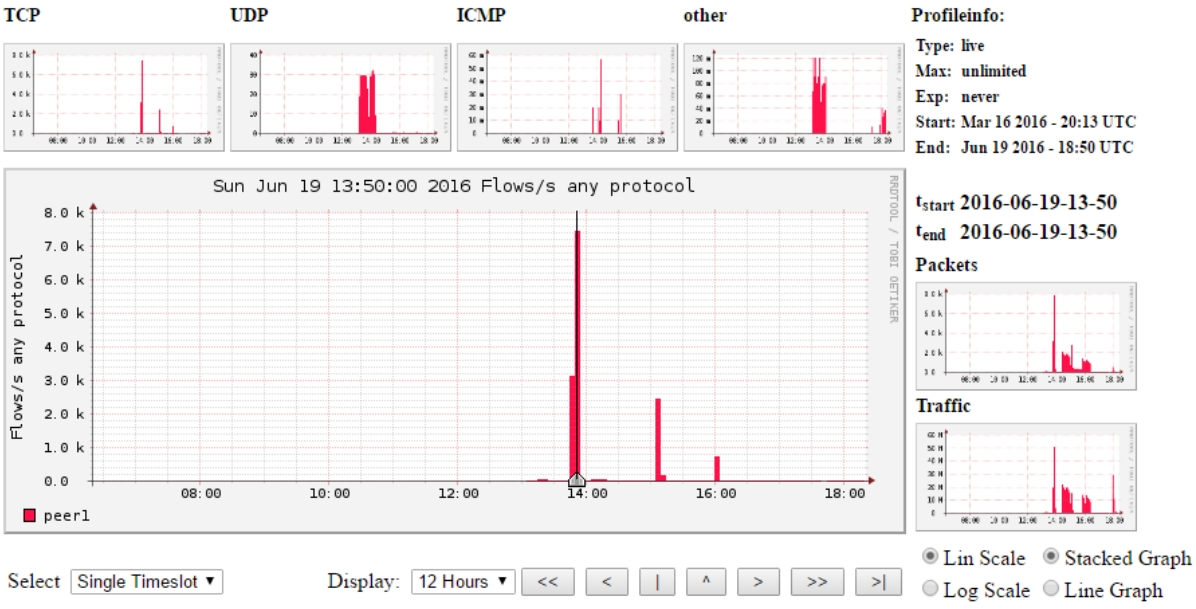
Na dalším obrázku 6.9 je zobrazen počet paketů/s a graf se do jisté míry liší, což je způsobeno velikostí jednotlivých paketů. V čase 15:50 vidíme, že je zde určitý objem paketů, nicméně počet

bitů/s je zde mizivý, tzn. že pakety měly malou velikost. Jejich řazení podle počtu přenesených paketů se opět liší od předchozího, viz. Obrázek 6.10.

```
** nfdump -M /data/nfsen/profiles-data/live/peer1  -T  -r 2016/06/19/nfcapd.201606191350 -n 100 -s ip/packets
nfdump filter:
any
Top 100 IP Addr ordered by packets:
Date first seen          Duration Proto           IP Addr    Flows(%)     Packets(%)       Bytes(%)         pps      bps   bpp
2016-06-19 13:46:55.000   408.189 any        192.168.1.68    2.2 M(99.9)    2.4 M(99.9)    1.9 G(99.8)     5796   37.2 M   801
2016-06-19 13:46:55.000   358.722 any      104.155.10.158        9( 0.0)    132214( 5.6)   197.2 M(10.4)     368    4.4 M  1491
2016-06-19 13:48:45.584   299.927 any        192.168.1.5       57( 0.0)     111( 0.0)     61617( 0.0)       0     1643   555
2016-06-19 13:49:36.515   250.242 any      192.168.1.179       52( 0.0)      52( 0.0)     69285( 0.0)       0     2214  1332
2016-06-19 13:48:45.584   289.571 any    239.255.255.250       11( 0.0)      50( 0.0)     16093( 0.0)       0      444   321
2016-06-19 13:48:54.863   296.097 any      192.168.1.110       46( 0.0)      46( 0.0)     61063( 0.0)       0     1649  1327
2016-06-19 13:48:53.281   288.192 any      192.168.1.188       43( 0.0)      43( 0.0)     56774( 0.0)       0     1576  1320
2016-06-19 13:49:01.039   274.062 any      192.168.1.251       41( 0.0)      41( 0.0)     54879( 0.0)       0     1601  1338
2016-06-19 13:48:47.137   285.236 any       192.168.1.70       40( 0.0)      40( 0.0)     53036( 0.0)       0     1487  1325
2016-06-19 13:48:52.043   286.898 any      192.168.1.191       40( 0.0)      40( 0.0)     54021( 0.0)       0     1506  1350
2016-06-19 13:48:48.622   301.217 any      192.168.1.242       39( 0.0)      39( 0.0)     52133( 0.0)       0     1384  1336
2016-06-19 13:49:08.341   266.860 any      192.168.1.206       36( 0.0)      38( 0.0)     50655( 0.0)       0     1518  1333
2016-06-19 13:48:58.677   275.112 any       192.168.1.86       35( 0.0)      37( 0.0)     48771( 0.0)       0     1418  1318
2016-06-19 13:49:11.978   276.729 any      192.168.1.144       35( 0.0)      35( 0.0)     46910( 0.0)       0     1356  1340
2016-06-19 13:48:50.382   288.962 any      192.168.1.243       34( 0.0)      35( 0.0)     46707( 0.0)       0     1293  1334
2016-06-19 13:49:57.837   222.617 any      192.168.1.167       34( 0.0)      34( 0.0)     45545( 0.0)       0     1636  1339
2016-06-19 13:48:50.852   198.594 any      192.168.1.151       33( 0.0)      33( 0.0)     44459( 0.0)       0     1790  1347
2016-06-19 13:48:49.036   290.209 any      192.168.1.171       33( 0.0)      33( 0.0)     44002( 0.0)       0     1212  1333
2016-06-19 13:49:22.881   257.364 any      192.168.1.205       33( 0.0)      33( 0.0)     44388( 0.0)       0     1379  1345
2016-06-19 13:49:33.451   256.388 any      192.168.1.246       33( 0.0)      33( 0.0)     44247( 0.0)       0     1380  1340
```

Obrázek 6-10: nfdump - textový výpis pakety/s

Na posledním grafu (obrázek 6.11) jsou vidět pouze ony DoS útoky. Jelikož je zde zobrazen počet toků/s můžeme s domnívat, že mimo tyto odchylky stanice mezi sebou komunikovaly, tudíž se nezvyšoval počet toků. Onen útok byl veden z náhodných stanic a byly odeslány pouze TCP/SYN pakety, tzn. že byl vytvořen vždy nový tok.



Obrázek 6-11: NfSen - tok/s

Na obrázku 6.12 vidíme jaký počet toku připadá na jednotlivé stanice. Stanice oběti má opět největší počet toků. Data, která jsem odchytil na stanici oběti a byla odeslána pomoci NetFlow mají velikost 73MB což není moc, protože to jsou jen hlavičky paketů bez dat samotných.

```
** nfdump -M /data/nfsen/profiles-data/live/peer1  -T  -r 2016/06/19/nfcapd.201606191350 -n 100 -s ip/flows
nfdump filter:
any
Top 100 IP Addr ordered by flows:
Date first seen          Duration Proto           IP Addr    Flows(%)     Packets(%)       Bytes(%)         pps      bps   bpp
2016-06-19 13:46:55.000   408.189 any        192.168.1.68    2.2 M(99.9)    2.4 M(99.9)    1.9 G(99.8)     5796   37.2 M   801
2016-06-19 13:48:45.584   299.927 any         192.168.1.5       57( 0.0)      111( 0.0)     61617( 0.0)        0     1643   555
2016-06-19 13:49:36.515   250.242 any       192.168.1.179       52( 0.0)       52( 0.0)     69285( 0.0)        0     2214  1332
2016-06-19 13:48:54.863   296.097 any       192.168.1.110       46( 0.0)       46( 0.0)     61063( 0.0)        0     1649  1327
2016-06-19 13:48:53.281   288.192 any       192.168.1.188       43( 0.0)       43( 0.0)     56774( 0.0)        0     1576  1320
2016-06-19 13:49:01.039   274.062 any       192.168.1.251       41( 0.0)       41( 0.0)     54879( 0.0)        0     1601  1338
2016-06-19 13:48:47.137   285.236 any        192.168.1.70       40( 0.0)       40( 0.0)     53036( 0.0)        0     1487  1325
2016-06-19 13:48:52.043   286.898 any       192.168.1.191       40( 0.0)       40( 0.0)     54021( 0.0)        0     1506  1350
2016-06-19 13:48:48.622   301.217 any       192.168.1.242       39( 0.0)       39( 0.0)     52133( 0.0)        0     1384  1336
2016-06-19 13:49:08.341   266.860 any       192.168.1.206       36( 0.0)       38( 0.0)     50655( 0.0)        0     1518  1333
2016-06-19 13:48:58.677   275.112 any        192.168.1.86       35( 0.0)       37( 0.0)     48771( 0.0)        0     1418  1318
2016-06-19 13:49:11.978   276.729 any       192.168.1.144       35( 0.0)       35( 0.0)     46910( 0.0)        0     1356  1340
2016-06-19 13:49:57.837   222.617 any       192.168.1.167       34( 0.0)       34( 0.0)     45545( 0.0)        0     1636  1339
2016-06-19 13:48:50.382   288.962 any       192.168.1.243       34( 0.0)       35( 0.0)     46707( 0.0)        0     1293  1334
2016-06-19 13:48:50.852   198.594 any       192.168.1.151       33( 0.0)       33( 0.0)     44459( 0.0)        0     1790  1347
2016-06-19 13:48:49.036   290.209 any       192.168.1.171       33( 0.0)       33( 0.0)     44002( 0.0)        0     1212  1333
2016-06-19 13:49:22.881   257.364 any       192.168.1.205       33( 0.0)       33( 0.0)     44388( 0.0)        0     1379  1345
2016-06-19 13:49:33.451   256.388 any       192.168.1.246       33( 0.0)       33( 0.0)     44247( 0.0)        0     1380  1340
2016-06-19 13:49:15.444   238.907 any         192.168.1.9       32( 0.0)       32( 0.0)     42614( 0.0)        0     1426  1331
2016-06-19 13:49:07.918   260.791 any       192.168.1.141       32( 0.0)       32( 0.0)     42676( 0.0)        0     1309  1333
```

Obrázek 6-12: NfSen - textový výpis tok/s

6.4 Suricata IDS

Suricata IDS nepodporuje čtení dat uložených ve formátu NetFlow, pro analýzu budeme používat formát pcap. Suricata IDS, jako taková nepodporuje ani analýzu anomálií učením, je tedy zapotřebí definovat statické anomálie nebo jak to Suricata IDS nazývá „Analýza anomálií protokolů“. Jedná se o sadu pravidel, kterými se určité protokoly vyznačují jako třeba číslo portu. V pravidle specifikujeme číslo portu a protokol aplikační vrstvy, a zda se mají rovnat nebo ne. V testovacím pravidle jsem použil následující syntaxi a pouze toto jedno pravidlo pro lepší orientaci v log souborech. Druhým útokem bylo skenování portů, což však v Suricata IDS nelze jednoduše implementovat a lze pouze detekovat přístup na konkrétní porty, ale ne celou sadu.

„alert tcp $HOME_NET any -> $HOME_NET 80 (msg:"HTTP DoS"; flags:S; count 100, seconds 5; sid:10001;rev:1;)“

Popis pravidla:

- „alert“ - nám určuje vygenerování alarmu, v případě porušení pravidla, další možností je „drop“, který daný paket zahodí,
- „tcp“ - definuje protokol transportní vrstvy,
- „$HOME_NET“ - definice podsítě, kterou definujeme v „suricata.yaml“. Můžeme definovat několik sítí, které pak při vícenásobném použití nemusíme stále dokola vypisovat,
- „any“, „80“ - definují čísla portů, pokud je „any“, znamená to jakýkoli port. Hodnoty před znakem „->” definují zdroj komunikace,
- „msg“ - zpráva, která se zobrazí při shodě paketu s tímto pravidlem,

- „flow“ - směr předpokládaného toku. Můžeme také definovat, možnost navázané komunikace
- „count“ - počet paketů, při jejichž překročení se pravidlo aplikuje,
- „flags“ - označení typu paketu, v našem případě to tedy je TCP/SYN paket.

Pomocí těchto pravidel Suricata IDS generuje alarmy či zahazuje pakety. Můžeme je upravit například pomocí negací, kdy se někdo bude snažit přistupovat na port 80 jiným než HTTP protokolem. V tomto případě přidáme před „ app-layer-protocol“ znak „!“.

Pokud bychom chtěli pravidlo rozšířit co nejobecněji na jakýkoli SYN DoS útok, vypadalo by takto:

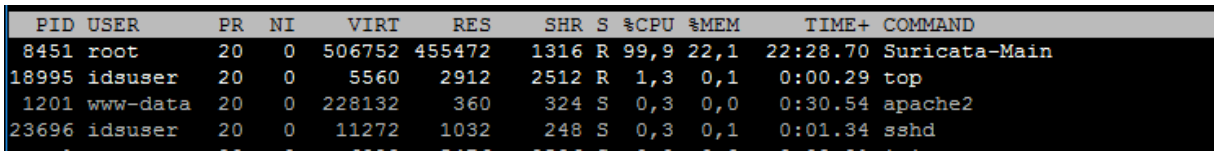
„*alert tcp any any -> any any (msg:"DoS"; flags:S; threshold:type both, track by_src, count 100, seconds 5; sid:10001;rev:1;)“*

Jelikož budeme analyzovat již odchycená data, spustíme Suricata IDS s přepínačem „-r“ což nám umožní číst uložený soubor ve formátu pcap. Odchycená data mají v tomto případě 5.7GB, to je v porovnání s NetFlow záznamy ze stejného provozu poměrně velký rozdíl.

„*sudo suricata -c /etc/suricata/suricata.yaml -r /data/utok.pcap“*

Pokud spustíme aplikaci pro čtení dat ze síťové karty v reálném čase, bude Suricata IDS postupně zapisovat alarmy do jednotlivých log souborů podle toho jak budou tyto události v čase probíhat.

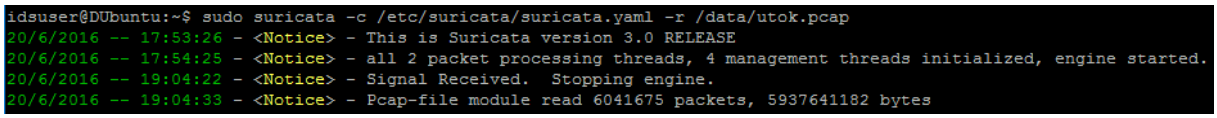
V prvním případě spustím Suricatu IDS v základním nastavení s úvodními pravidly do kterých nebudu zasahovat. Stejně jako v případě čtení dat ze síťového rozhraní v reálném čase i zde jsou log soubory průběžně aktualizovány. Analýza naštěstí netrvá tak dlouho, jako doba skutečně odchyceného provozu, ale čtyřhodinový provoz byl analyzován za hodinu a pět minut na stroji s omezenou kapacitou. Vzhledem k tomu, že jsem použil úvodní pravidla, je zde mnoho „falešně pozitivních“ alarmů (tisíce), kterými se musíme probrat, abychom se dopátrali skutečného útoku. Je tedy lepší upravit si tato pravidla, aby co nejlépe odpovídali struktuře naší sítě. Pro druhou analýzu tedy použijeme pouze pravidlo, které jsme si uvedli v úvodu. Upozorňuji, že to není vhodné do běžného provozu z důvodu rozmanitosti a promyšlenosti útoků. Pokud se podíváme na vytížení procesoru během analýzy dat, jistě nás nepřekvapí že Suricata IDS využívá téměř 100%, samozřejmě tato hodnota kolísá a vidíme ji na obrázku 6.13.



PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
8451	root	20	0	506752	455472	1316	R	99,9	22,1	22:28.70	Suricata-Main
18995	idsuser	20	0	5560	2912	2512	R	1,3	0,1	0:00.29	top
1201	www-data	20	0	228132	360	324	S	0,3	0,0	0:30.54	apache2
23696	idsuser	20	0	11272	1032	248	S	0,3	0,1	0:01.34	sshd

Obrázek 6.13: Suricata IDS - vytížení procesoru a RAM

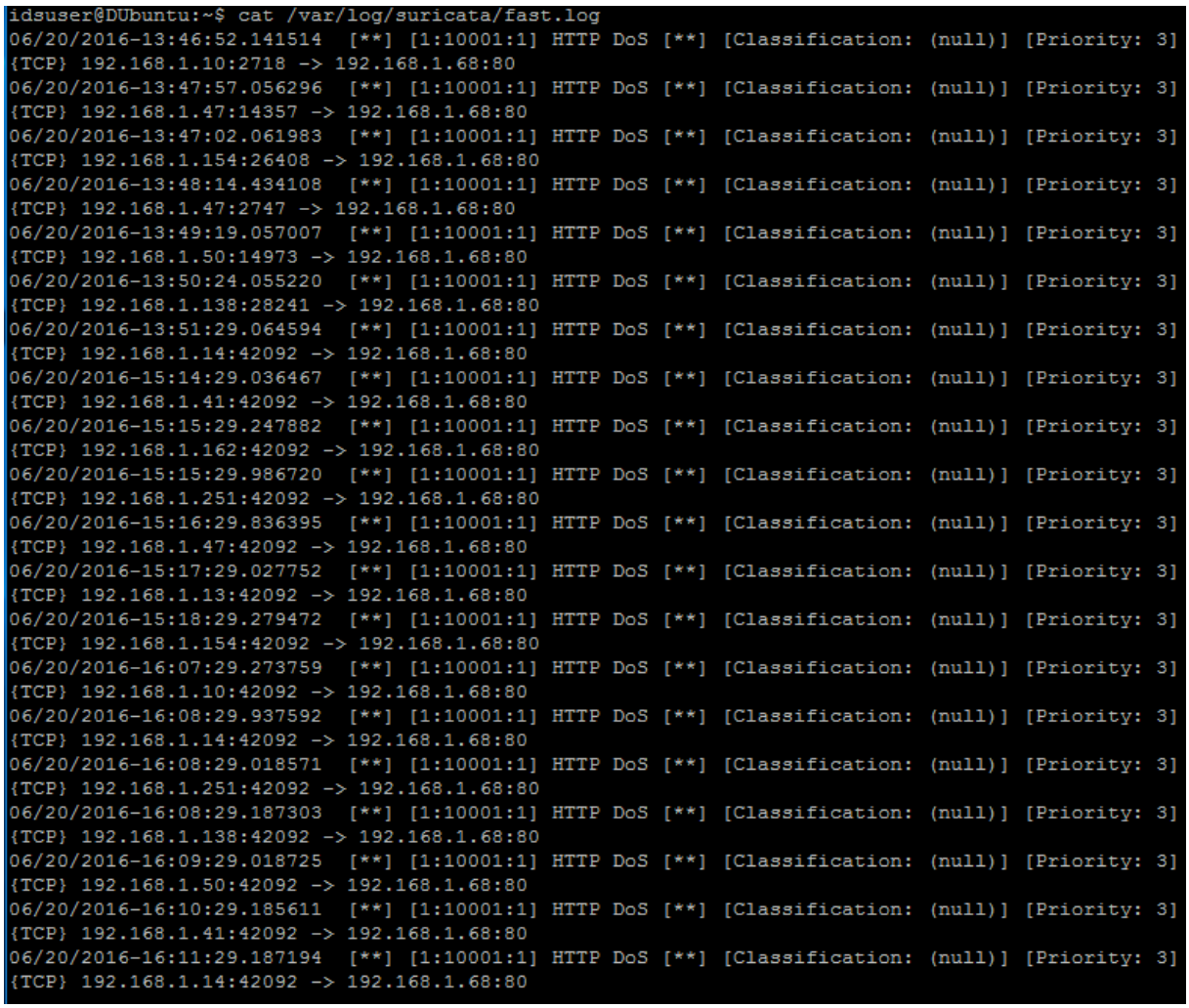
Běh Suricaty IDS jako takový při analýze pouze zobrazí, že je spuštěný a na konci zobrazí, kolik paketů a bajtů analyzoval jak je uvedeno na obrázku 6.14.



Obrázek 6.14: Suricata IDS - běh programu

Zajímavější jsou však log soubory a jejich velikosti, které jsou v případě „eve.json“ souboru, který představuje způsob jakým loguje NetFlow, přes 2GB. V případě „fast.log“ je to 1.5GB a ostatní jsou zanedbatelné vůči těmto. V této práci se nebudu zabývat jejich obsahem. Suricata IDS spustíme ještě jednou a to pouze s jedním pravidlem, kde výpis bude jistě omezenější a relevantnější.

V případě použití pouze jednoho pravidla trvá analýza pouze 6 minut, což je neporovnatelný rozdíl vůči všem v předchozím případě. Soubor „eve.json“ je stejně velký, tudíž má 2GB, počet paketů pro zobrazení v NetFlow formátu je stejný. Nicméně soubor „fast.log“ má velikost pouze 2.8kB aobsahuje 20 záznamů, které nám hlásí možný DoS útok jak je vidět na obrázku 6.15.



Obrázek 6-15: Suricata IDS - zobrazení alarmů

Z výpisu vidíme, že ve stejných časech jako v případě NfSen byly výchylky z běžného provozu, jsou i zde ve stejných časech zobrazeny potencionální DoS útoky. Tyto alarmy jsou vygenerovány při porovnání celých 5.7GB provozu, za každé porušení pravidla jeden záznam. Záznamy mají zde specifikován přesný čas a datum porušení specifikace pravidla, protokol a IP adresa s portem, ze které byl tento útok veden a samozřejmě IP adresa a port oběti. Vidíme, že IP útočníka

se mění tak, jak jsme specifikovali při útoku, nelze tedy jednoznačně určit kdo útočil. Jediné co můžeme zjistit je tedy to, že někdo se pokouší provést TCP/SYN DoS útok na naše systémy. Pokud bychom vytvořili pravidlo pro hloubkovou kontrolu, která může zjistit jestli data v paketu se schodují s daty specifikovanými v pravidle a na základě nich se dozvíme více informací o úmyslech útočníka. Můžeme specifikovat jak SSH klíč, se kterým je možno přihlásit se do systému a při jeho nerovnosti vygenerovat záznam, dále například typ či verzi prohlížeče web serveru, přičemž tyto alarmy budou generovány do souboru „http.log“.

Poslední log soubor jsou statistiky odchycených paketů, jsou zde uváděny počty zpracovaných paketů celkem za celou dobu provozu Suricata IDS nebo z celého zpravovaného souboru odchyceného provozu. Při pohledu na obrázek 6.16 jsou nejzajímavější řádky „tcp.syn“ a „tcp.synack“. Vidíme, že počet „tcp.syn“ je mnohonásobně vyšší než „tcp.synack“ a to víc než 130 tisíc krát. Z toho můžeme usuzovat, že příjemce nestíhá odpovídat na požadavky TCP/SYN a je pravděpodobně zahlcen.

```
-------------------------------------------------------------------
Date: 6/20/2016 -- 21:49:22 (uptime: 0d, 00h 00m 48s)
-------------------------------------------------------------------
Counter                   | TM Name                   | Value
-------------------------------------------------------------------
decoder.pkts              | Total                     | 564480
decoder.bytes             | Total                     | 453705983
decoder.ipv4              | Total                     | 563052
decoder.ipv6              | Total                     | 773
decoder.ethernet          | Total                     | 564480
decoder.tcp               | Total                     | 532311
decoder.udp               | Total                     | 31433
decoder.icmpv4            | Total                     | 10
decoder.avg_pkt_size      | Total                     | 803
decoder.max_pkt_size      | Total                     | 14546
tcp.sessions              | Total                     | 273758
tcp.invalid_checksum      | Total                     | 82
tcp.syn                   | Total                     | 273758
tcp.synack                | Total                     | 2
tcp.rst                   | Total                     | 258438
flow_mgr.closed_pruned    | Total                     | 262
flow_mgr.new_pruned       | Total                     | 133497
flow_mgr.est_pruned       | Total                     | 232
flow.spare                | Total                     | 10453
tcp.memuse                | Total                     | 286720
tcp.reassembly_memuse     | Total                     | 12244864
flow.memuse               | Total                     | 54864436
idsuser@DUbuntu:~$
```

Obrázek 6-16: Suricata IDS - statistiky

6.5 Bro IDS

Bro IDS stejně jako Suricata IDS nepodporuje behaviorální analýzu dat. Je tedy nutné použít analýzu statickou. Bro IDS nepodporuje nativně ani NetFlow, je nutné doinstalovat nástroj *„nfcollector a ftwire2to“* pro čtení NetFlow dat ze síťové karty a jejich převod na formát používaný

v Bro IDS. Nicméně tento převod znehodnotí obsah dat a analýza pomocí Bro IDS je zbytečné plýtvání možností tohoto nástroje. Proto tento převod nebudu provádět.

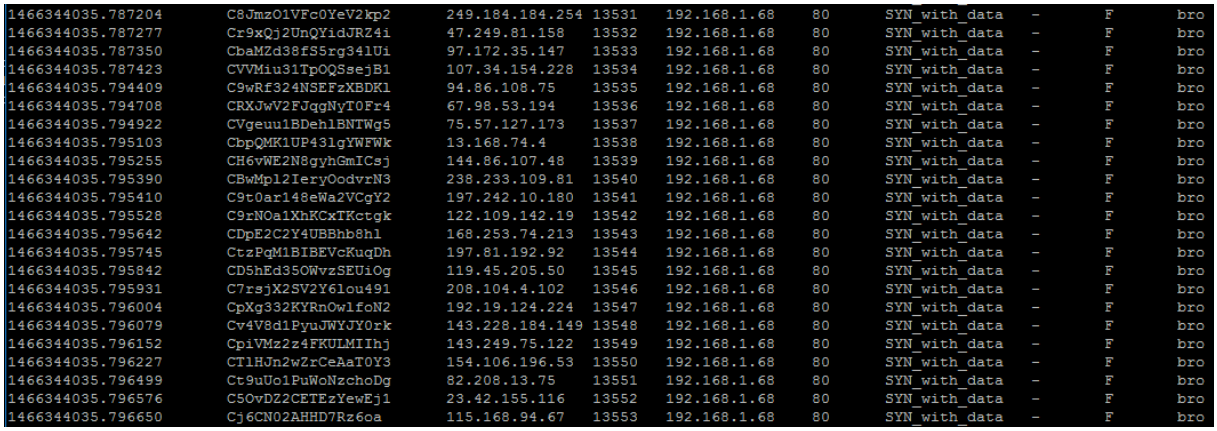
Bro IDS použiji v režimu analýzy souboru „utok.pcap“ a pro tento účel využiji příkaz „*sudo bro -r /data/utok.pcap“*. Po spuštění příkazu musíme opět čekat, než se analyzuje celý provoz. Na rozdíl od Suricaty IDS, Bro IDS produkuje mnohem více log souborů, které jsou rozděleny jednak časově a také podle dalších kriterií. Můžeme zde najít log soubory od komunikace obecně, přes DNS komunikaci až po log soubor nazvaný "weird", ten obsahuje všechen provoz, který není obvyklý pro běžnou komunikaci pomocí standardních protokolů. Postupně jak Bro IDS běží, vytváří nové log soubory na základě toho, kdy k jakému porušení pravidel dojde. Z počátku je složitější se v těchto souborech vyznat, ale posléze jsou velmi přehledné a užitečné. Pro útoky, které jsme aplikovali, použijeme především dva skripty a to, „scan.bro“ a „synflood.bro“ . Oba jsou dostupné v základní verzi Bro IDS. Upravíme si je však dle svých potřeb snížením počtu paketů pro úspěšný DoS útok a dalšími drobnostmi, které byli provedeny v „synflood.bro“. Druhým skriptem byl „scan.bro, který jsem upravil pro sken menších rozměrů. Jak je vidět na obrázku 6.17 Bro IDS vytežuje procesor mnohem méně než Suricata IDS, ale za to RAM je vytížena dostatečně. Zpracování a analýza dat v tomto případě trvala víc než hodinu a půl, což je o něco více než v případě Suricata IDS se všemi nativními pravidly. Dostali jsme se zde také na mnohem menší velikost log souborů, kdy jednotlivé soubory nepřekročili 200 MB.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
16070	root	20	0	1996000	1,711g	1512	D	22,9	86,9	22:29.63	bro
2316	guest-K+	20	0	326044	31860	0	S	0,3	1,5	0:52.30	compiz
15375	idsuser	20	0	12640	812	604	S	0,3	0,0	0:10.64	sshd
16117	root	20	0	0	0	0	S	0,3	0,0	0:00.58	kworker/0:0
1	root	20	0	4580	488	404	S	0,0	0,0	0:07.90	init

Obrázek 6.17: Bro IDS - vytížení procesoru a RAM

DoS útok pomocí Bro IDS detekujeme tak, že nahlédneme do log souboru „weird.log“, obrázek 6.18, zde vidíme IP adresu a port útočníka, IP adresu a port oběti a důležitou informaci „SYN_with_data“, to znamená, že byl přijat TCP/SYN, který obsahoval data. Když si vzpomenete jak jsme útočili, posílali jsme TCP/SYN paket a 1400 bajtů dat. Velký objem dat je zde posílán pro co největší zahlcení oběti.

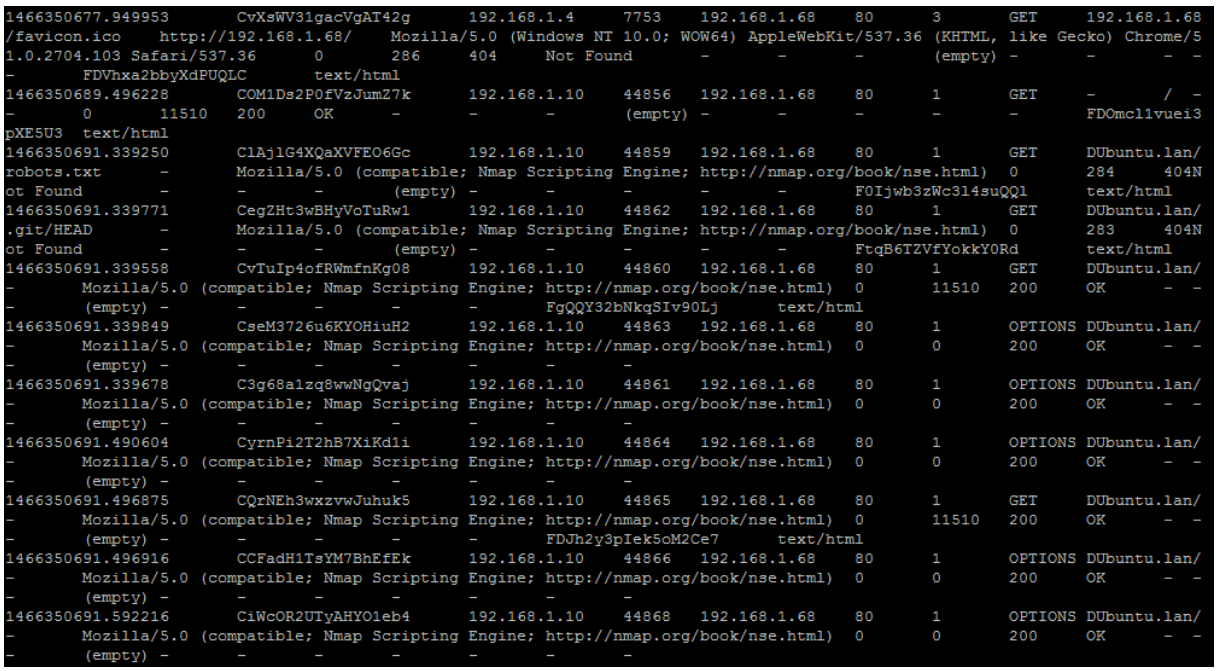
V případě druhého útoku, tedy skenování portů, jsem nebyl schopen jej detekovat. Nejsem si jistý, zda je to z důvodu typu a intenzity provedeného skenování nebo špatně nastaveného skriptu pro detekci. Pokoušel jsem se jej několikrát měnit, ale v žádném případě nebylo vygenerování varování o možném skenování. Jediné co bylo možné zjistit je, že útočník se pokoušel zjistit jaké parametry má web a SSH serve,r což byli jediné dva otevřené porty. Tyto informace jsou v souboru „http.log“ a „ssh.log“. Hlavním vodítkem pro zjištění které informace se snažil někdo získat pomocí skenování je, že záznam obsahuje klíčové slovo Nmap, což je skenovací nástroj, který jsem použil.



```
1466344035.787204       C8JmzO1VFc0YeV2kp2      249.184.184.254 13531   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.787277       Cr9xQj2UnQYidJRZ4i      47.249.81.158   13532   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.787350       CbaMZd38fS5rg341Ui      97.172.35.147   13533   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.787423       CVVMiu31TpOQSsejB1      107.34.154.228  13534   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.794409       C9wRf324NSEFzXBDK1      94.86.108.75    13535   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.794708       CRXJwV2FJqgNyT0Fr4      67.98.53.194    13536   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.794922       CVgeuu1BDehlBNTWg5      75.57.127.173   13537   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795103       CbpQMK1UP43lgYWFWk      13.168.74.4     13538   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795255       CH6vWE2N8gyhGmICsj      144.86.107.48   13539   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795390       CBwMpl2IeryOodvrN3      238.233.109.81  13540   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795410       C9t0ar148eWa2VCgY2      197.242.10.180  13541   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795528       C9rNOa1XhKCxTKctgk      122.109.142.19  13542   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795642       CDpE2C2Y4UBBhb8hl       168.253.74.213  13543   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795745       CtzPqM1BIBEVcKuqDh      197.81.192.92   13544   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795842       CD5hEd35OWvzSEUiOg      119.45.205.50   13545   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.795931       C7rsjX2SV2Y6lou491      208.104.4.102   13546   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796004       CpXg332KYRnOwlfoN2      192.19.124.224  13547   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796079       Cv4V8d1PyuJWYJY0rk      143.228.184.149 13548   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796152       CpiVMz2z4FKULMIIhj      143.249.75.122  13549   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796227       CTlHJn2wZrCeAaT0Y3      154.106.196.53  13550   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796499       Ct9uUo1PuWoNzchoDg      82.208.13.75    13551   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796576       C5OvDZ2CETEzYewEj1      23.42.155.116   13552   192.168.1.68    80      SYN_with_data   -       F       bro
1466344035.796650       Cj6CN02AHHD7Rz6oa       115.168.94.67   13553   192.168.1.68    80      SYN_with_data   -       F       bro
```

Obrázek 6.18: Bro IDS - "weird.log"

V souboru „http.log“ je více záznamů, ale pokus o skenování poznáme tak, že obsahuje klíčové slovo Nmap. Je zde uvedena především útočníkova IP adresa (192.168.1.10) a typ prohlížeče, se kterým jsem se snažil poslat několik zpráv pro zjištění více informací o web serveru a bylo to především „GET“ a „OPTIONS“, kdy server odpověděl se zprávou „200 OK“. Každý záznam se označuje číslem na začátku řádku. Zkrácený výpis „http.log“ viz. Obrázek 6.19. Na prvním řádku vidíme legitimní požadavek na webový server, který obsahuje podobné informace jako skenování. Navíc je zde uvedena URL adresa, na niž se uživatel snažil připojit a samozřejmě neobsahuje záznam o nástroji Nmap.

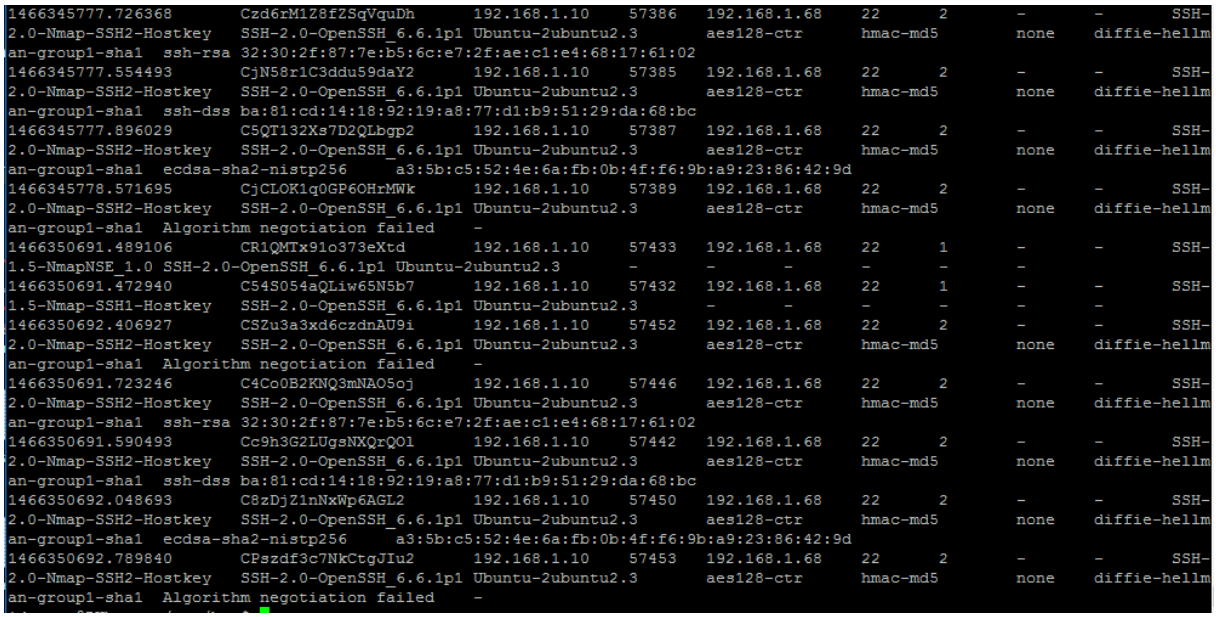


```
1466350677.949953       CvXsWV31gacVgAT42g      192.168.1.4     7753    192.168.1.68    80      3       GET     192.168.1.68
/favicon.ico    http://192.168.1.68/    Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/5
1.0.2704.103 Safari/537.36      0       286     404     Not Found       -       -       -       (empty) -       -       -  -
-       FDVhxa2bbyXdPUQLC       text/html
1466350689.496228       COM1Ds2P0fVzJumZ7k      192.168.1.10    44856   192.168.1.68    80      1       GET     -       /  -
-       0       11510   200     OK      -       -       -       (empty) -       -       -       -       -       FDOmcl1vuei3
pXE5U3  text/html
1466350691.339250       ClAjlG4XQaXVFEO6Gc      192.168.1.10    44859   192.168.1.68    80      1       GET     DUbuntu.lan/
robots.txt      -       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       284     404N
ot Found        -       -       -       (empty) -       -       -       -       -       FOIjwb3zWc3l4suQQl      text/html
1466350691.339771       CegZHt3wBHyVoTuRw1      192.168.1.10    44862   192.168.1.68    80      1       GET     DUbuntu.lan/
.git/HEAD       -       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       283     404N
ot Found        -       -       -       (empty) -       -       -       -       -       FtqB6TZVfYokkY0Rd       text/html
1466350691.339558       CvTuIp4ofRWmfnKg08      192.168.1.10    44860   192.168.1.68    80      1       GET     DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       11510   200     OK      -  -
-       (empty) -       -       -       -       -       -       FgQQY32bNkqSIv90Lj      text/html
1466350691.339849       CseM3726u6KYOHiuH2      192.168.1.10    44863   192.168.1.68    80      1       OPTIONS DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       0       200     OK      -  -
-       (empty) -       -       -       -       -       -       -
1466350691.339678       C3g68a1zq8wwNgQvaj      192.168.1.10    44861   192.168.1.68    80      1       OPTIONS DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       0       200     OK      -  -
-       (empty) -       -       -       -       -       -       -
1466350691.490604       CyrnPi2T2hB7XiKd1i      192.168.1.10    44864   192.168.1.68    80      1       OPTIONS DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       0       200     OK      -  -
-       (empty) -       -       -       -       -       -       -
1466350691.496875       CQrNEh3wxzvwJuhuk5      192.168.1.10    44865   192.168.1.68    80      1       GET     DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       11510   200     OK      -  -
-       (empty) -       -       -       -       -       -       FDJh2y3pIek5oM2Ce7      text/html
1466350691.496916       CCFadH1TsYM7BhEfEk      192.168.1.10    44866   192.168.1.68    80      1       OPTIONS DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       0       200     OK      -  -
-       (empty) -       -       -       -       -       -       -
1466350691.592216       CiWcOR2UTyAHYO1eb4      192.168.1.10    44868   192.168.1.68    80      1       OPTIONS DUbuntu.lan/
-       Mozilla/5.0 (compatible; Nmap Scripting Engine; http://nmap.org/book/nse.html)  0       0       200     OK      -  -
-       (empty) -       -       -       -       -       -       -
```

Obrázek 6.19: Bro IDS - "http.log"

Pokus o skenování SSH serveru je v souboru „ssh.log“, kde najdeme opět klíčové slovo Nmap, v tomto případě u všech záznamů, protože zde nebyl žádný legitimní požadavek o přístup přes protokol SSH. Je zde zaznamenána IP adresa a port oběti. Jde samozřejmě o port 22, dále zde najdeme velice důležitou informaci a tou je IP adresa útočníka. Log soubor také obsahuje, jaké verze

SSH útočník zkoušel, jakého klienta pro přístup používá, jaká je verze a typ SSH serveru, typ šifry pro přenos zpráv, typ hašovacího algoritmu pro generování klíčů, jaký protokol SSH server používá pro výměnu klíčů, typ algoritmu pro generování klíčů a nakonec onen veřejný klíč. Vše lze vidět na obrázku 6.20



Obrázek 6.20: Bro IDS - "ssh.log"

Bro IDS generuje mnohem více log souborů v kterých se dozvíme téměř veškeré informace o dění na síti. Tyto log soubory jsou velmi obsáhlé a můžeme se dozvědět, jaký prohlížeč uživatel používá či jaký typ web serveru je nainstalován na serveru. Dále můžeme zjistit jaká data se uživatel snažil přenést a pomocí jakých protokolů.

7 Závěr

V této diplomové práci jsme se dozvěděli, jakým způsobem se dá analyzovat provoz na síti, ať už v reálném čase nebo později z odchycených dat. Dle mého názoru je nejvýhodnější provozovat kontinuální záznam aktivit na síti pomocí NetFlow a v případě nějaké neočekávané události či aktivity spustit paketovou analýzu s možným záznamem do pcap souboru, nejlépe automaticky. Následně či okamžitě můžeme mnohem intenzivněji data z onoho pcap souboru analyzovat a zjistit více informací o oné aktivitě. Tímto způsobem ušetříme hodně místa pro ukládání záznamů, ale také získáme výhody paketové analýzy. V našem případě byl rozdíl objemu dat obrovský. V případě datových toků to bylo 73 MB za 4 hodiny provozu což je cca 160 GB dat za rok a není problém s jejich uchováním. V případě ukládání veškerého provozu pro paketovou analýzu jsme se dostali na 5.7 GB dat, což by nám za rok dělalo cca 12.5 TB dat, která by se také dala v dnešní době již bez problému uložit. Představme si, že jsme velká firma, která má mnohem vetší toky dat než byli v naší simulaci a může se dostat až na stovky nebo tisíce peta bajtů dat.

Díky nástroji NfSen a malému objemu NetFlow jsme se dozvěděli základní informace o komunikaci mezi jednotlivými stanicemi na síti a byli jsme schopni identifikovat odchylky od normálního provozu. Nástroje Suricata IDS a Bro IDS jsou již mnohem komplexnější, jelikož nepodporují zpracování NetFlow dat, použili jsme pro analýzu pcap soubory, které obsahují záznam o veškerém provozu na sítí, zjistili jsme se mnohem obsáhlejší informace o komunikaci na síti. Suricata je poměrně mladý nástroj, s nímž není složité pracovat či vytvářet pravidla. Pravidla jsou vytvořena pomocí klíčových slov a jsou jednoduše aplikovatelná. Na jejich základě se vygeneruje alarm o porušení tohoto pravidla nebo se paket rovnou zahodí a případně se i tomto zahození vygeneruje alarm. Bro IDS je naproti tomu již vyspělý nástroj a tvorba pravidel je mnohem složitější a je potřeba znát skriptovací jazyk Bro vytvořený přesně pro tento účel. Díky němu jsme schopní detekovat mnohem složitější útoky či provádět hloubkovější analýzu. Je tedy vhodnější pro pokročilejší uživatele. Nicméně pokud jej pochopíme a jsme schopni pro něj psát skripty považoval bych tento nástroj za momentálně lepší.

8 Použitá literatura

1. NORTHCUTT, Stephen. *Bezpečnost sítí: velká kniha*. Brno: CP Books, 2005. Security (CP Books). ISBN 80-251-0697-7.
2. Libpcap File Format [online]. 2015 [cit. 2016-06-26]. Dostupné z: https://wiki.wireshark.org/Development/LibpcapFileFormat
3. Pcap File Format [online]. 2012 [cit. 2016-06-26]. Dostupné z: http://www.kroosec.com/2012/10/a-look-at-pcap-file-format.html
4. Network Media [online]. 2010 [cit. 2016-06-26]. Dostupné z: http://wiki.wireshark.org/CaptureSetup/NetworkMedia
5. Link Types [online]. 2010 [cit. 2016-06-26]. Dostupné z: http://www.tcpdump.org/linktypes.html
6. File Signatures [online]. [cit. 2016-06-26]. Dostupné z: http://www.filesignatures.net/index.php?page=all
7. Pcapng [online]. 2004 [cit. 2016-06-26]. Dostupné z: https://www.winpcap.org/ntar/draft/PCAP-DumpFileFormat.html
8. Wireshark [online]. 2004-2014 [cit. 2016-06-26]. Dostupné z: https://www.wireshark.org/docs/wsug_html/#ChIntroWhatIs
9. NetFlow Version 9 Flow-Record Format [online]. 2011 [cit. 2016-06-26]. Dostupné z: http://www.cisco.com/en/US/technologies/tk648/tk362/technologies_white_paper09186a00800a3db9.html
10. Introduction to Cisco IOS NetFlow [online]. 2012 [cit. 2016-06-26]. Dostupné z: http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html
11. NetFlow Services Solutions Guide [online]. 2001 [cit. 2016-06-26]. Dostupné z: http://www.cisco.com/c/en/us/td/docs/ios/solutions_docs/netflow/nfwhite.html
12. NfSen [online]. 2011 [cit. 2016-06-26]. Dostupné z: http://nfsen.sourceforge.net/
13. NfDump [online]. 2014 [cit. 2016-06-26]. Dostupné z: http://nfdump.sourceforge.net/
14. Suricata IDS [online]. 2009 [cit. 2016-06-26]. Dostupné z: https://oisf.net/about-us/
15. Suricata IDS [online]. 2006-2015 [cit. 2016-06-26]. Dostupné z: https://redmine.openinfosecfoundation.org/projects/suricata/wiki/What_is_Suricata
16. Suricata IDS vlastnosti [online]. 2006-2015 [cit. 2016-06-26]. Dostupné z: https://suricata-ids.org/features/all-features/
17. Suricata IDS instalace [online]. 2011 [cit. 2016-06-26]. Dostupné z: https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Ubuntu_Installation
18. Suricata IDS nastavení [online]. 2006-2015 [cit. 2016-06-26]. Dostupné z: https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Basic_Setup
19. Bro IDS [online]. 2016 [cit. 2016-06-26]. Dostupné z: https://www.bro.org/sphinx/intro/

20. Bro IDS manuál [online]. 2004 [cit. 2016-06-26]. Dostupné z: http://www.gnu-darwin.org/www001/src/ports/security/bro/work/bro-1.2.1/doc/user-manual/Bro-user-manual.pdf

21. Bro IDS instalace [online]. 2016 [cit. 2016-06-26]. Dostupné z: https://www.bro.org/sphinx/install/install.html

22. Fprobe [online]. 2010 [cit. 2016-06-26]. Dostupné z: http://manpages.ubuntu.com/manpages/xenial/en/man8/fprobe.8.html

23. MCCLURE, Stuart, Joel SCAMBRAY a George KURTZ. Hacking bez tajemství: velká kniha. 3. aktualiz. vyd. Brno: Computer Press, 2003. Security (CP Books). ISBN 80-722-6948-8.